计算机组成原理

计算机组成原理课程组

（刘旭东、高小鹏、肖利民、牛建伟、栾钟治）

第三部分：时序逻辑电路设计

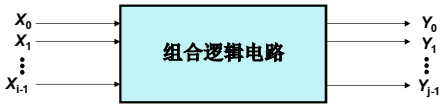
一. 锁存器和触发器
1. SR/D锁存器
2. D触发器
3. JK触发器

二. 有限状态机
1. Moore型有限状态机
2. Mealy型有限状态机

三. 时序逻辑电路设计分析
1. 数据寄存器
2. 移位寄存器
3. 计数器
4. Verilog HDL 设计
5. 时序电路的时序

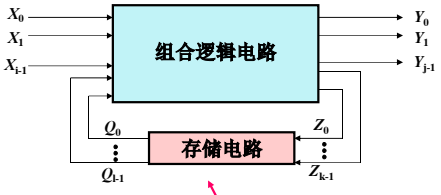
组合逻辑电路的不足

- 组合逻辑电路的特点
  - 电路输出端的状态完全由输入端的状态决定，不受系统中时钟脉冲的控制
  - 是一种无记忆电路——输入信号消失，则输出信号也会立即消失
- 在数字系统中，有时需要将参与（算术或逻辑）运算的数 据和运算结果保存起来——在组合逻辑电路的输出端需要 具有记忆功能的部件
- 触发器（Flip-Flop，FF）就是构成记忆功能部件的基本单 元，或者说是实现存储（记忆）功能的基本单元电路。



时序逻辑电路的特点

- 当时的输出由当时的输入与电路的原来状态决定——具有“记忆”功能
- 结构特点：由组合逻辑电路和存储电路构成



触发器（Flip-Flop，FF）或寄存器

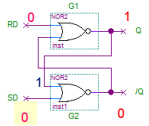
触发器的特点与分类

❖**触发器**是一种有记忆功能的器件，是构成时序逻辑电路的基本器件

❖特点：

Q称为状态变量

两个稳定的状态—
—双稳态触发器



1.有两个互非的输出 Q和$\overline{Q}$，

当$Q=0(\overline{Q}=1)$时称为0态，当 $Q=1(\overline{Q}=0)$时称为1态；

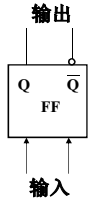
2.无外加信号作用时触发 器保持原来状态（原态 ）不变 — —具有记忆功能，n级触发器可以记忆 n位二进制信息的2^n种状态；

3.在外加信号的作用（触 发）下，触发器可以改 变原态（具有置0和置1功能）。

Q^n（原态）$\longrightarrow Q^{n+1}$(次态）

触发器原来的状态

触发器改变后的状态



基本RS触发器

❖ 基本RS锁存器可以自行保持输出状态，是各种触发器的基本构成部分

❖ 基本RS触发器可以用与非门或者或非门构成

❖ RS：Reset/Set

❖ 功能

（1）保持功能 $\overline{R}_D=1,\overline{S}_D=1$

触发器保持原来的状态不变

（2）置0功能 $\overline{R}_D=0,\overline{S}_D=1$

触发器的次态变为0

（3）置1功能 $\overline{R}_D=1,\overline{S}_D=0$

触发器的次态变为1

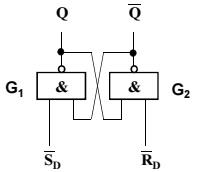
❖约束条件 $\overline{R}_D+\overline{S}_D=1$

非号，低有效

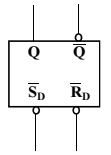
$\overline{R}_D$、$\overline{S}_D$不能同时为0

D: Direct

输入信号直接控制触发器的输出



电路结构



逻辑符号

触发器逻辑功能的表示方法

❖ 触发器的逻辑功能可以用功能表、真值表（特性表）、特性方程、状态转换图和时序图等来表示。

（1）功能表

$\overline{S}_D\overline{R}_DQ^n$	Q^{n+1}	功能
1 1 0	0	保持
1 1 1	1	
1 ↓ 0	0	置0
1 ↓ 1	0	
↓ 1 0	1	置1
↓ 1 1	1	
0 0 0	X	不确定
0 0 1	X	

（2）真值表(特性表)

$\overline{S}_D$ $\overline{R}_D$ Q^n	Q^{n+1}
0 0 0	x
0 0 1	x
0 1 0	1
0 1 1	1
1 0 0	0
1 0 1	0
1 1 0	0
1 1 1	1

❖将原态也作为一个变量列入了真值表，将这种含有状态变量的真值表称为特性表

❖特性表：电路输出次态与原态以及输入之间功能关系的表格

基本RS触发器的特性方程

❖特性方程：反映触发器次态与原态以及输入之间功能关系的函数表达式。

➢由特性表利用最小项推导法推导得出（利用约束条件化简）；

$$Q^{n+1}=\overline{\overline{S}}_D\overline{R}_D\overline{Q}^n+\overline{\overline{S}}_D\overline{R}_DQ^n+\overline{S}_D\overline{R}_DQ^n$$
$$=\overline{\overline{S}}_D\overline{R}_D+\overline{S}_D\overline{R}_DQ^n=S_D\overline{R}_D+\overline{S}_D\overline{R}_DQ^n$$

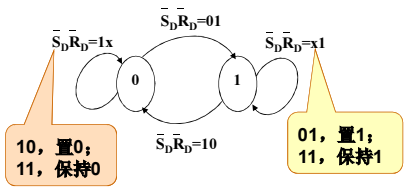
（3）特性方程

$$Q^{n+1}=S_D+\overline{R_D}\cdot Q^n$$
$$\overline{S_D}+\overline{R_D}=1\text{或}S_D\cdot R_D=0$$
（约束条件）

状态转换图

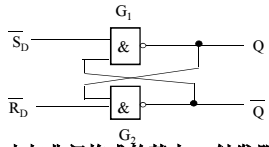
（4）状态转换图

- 简称状态图，是用来表示触发器状态变化（转移）的图形
- 用圆圈表示0或1状态，用带箭头的线表示状态变化的方向，线上的数据表示状态变化需要的输入条件。



基本RS触发器的HDL设计

- 结构描述方式
  - 根据电路结构写出输出信号的逻辑表达式；
  - 采用assign语句描述



$$Q = \overline{S_D \cdot Q}$$

$$\overline{Q} = \overline{R_D \cdot Q}$$

由与非门构成的基本RS触发器

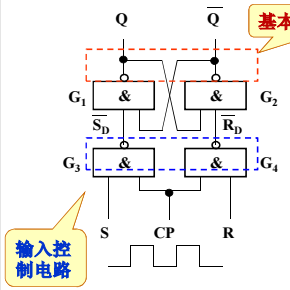
```
module RS_FF(Q,QN,SDN,RDN);
    input   SDN,RDN;
    output  Q,QN;
    assign  Q = !(SDN && QN);
    assign  QN= !(RDN && Q);
endmodule
```

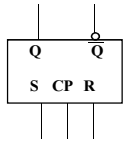

钟控RS锁存器

- 在数字系统中，为了协调各部分电路的运行，常常要求某些触发器在时钟信号的控制下同时动作，这就需要增加一个控制端（时钟），只有在控制端作用脉冲时触发器才能动作。
- 这种有时钟控制端的触发器叫做钟控锁存器。
- 由于这里时钟信号为高电位（或低电位）时触发器的状态随输入变化，所以钟控锁存器是电位触发方式的触发器。
- 钟控锁存器在时钟控制下同步工作，所以也称为同步锁存器。

钟控RS锁存器的工作原理

- 电路结构和逻辑符号





- 电位触发：在控制电位CP的控制下接收数据。
- CP＝0时，$\bar{S}_D$=$\bar{R}_D$=1，保持原有状态。
- 当CP＝1时，其输出状态由R、S端的输入信号决定——具有基本RS触发器的功能。
- 电位触发有正电位触发和负电位触发。

钟控RS锁存器的逻辑功能表示

- CP=0时，锁存器处于保持状态；
- CP=1 时，具有基本RS触发器的功能——称为钟控RS锁存器

特性表(CP=1)

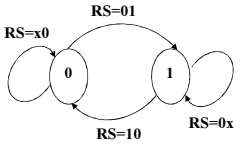
S	R	Q^n	Q^{n+1}	功能
0	0	0	0	保持
0	0	1	1	
0	1	0	0	置0
0	1	1	0	
1	0	0	1	置1
1	0	1	1	
1	1	0	X	不确定
1	1	1	X	

R、S不能同时为1

特性方程

$$Q^{n+1} = S + \overline{R} \cdot Q^n$$

$$S \cdot R = 0 \text{（约束条件）}$$

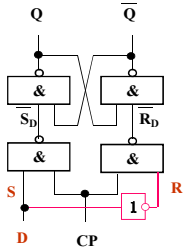


$$Q^{n+1} = \overline{S}RQ^n + (S\overline{R}Q^n + S\overline{R}Q^n) = \overline{S}RQ^n + S\overline{R}$$

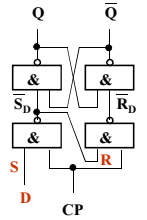
状态转换图

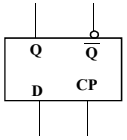
钟控D锁存器

- 问题——如何消除钟控RS锁存器的不确定状态？
  - 将钟控RS锁存器的输入由R、S双端输入改为单端输入（D）——即将其S输入端改为D输入端，然后经过非门接R端——S、R总是互非，钟控D锁存器不会出现不定状态！
- 钟控D锁存器电路结构和逻辑符号

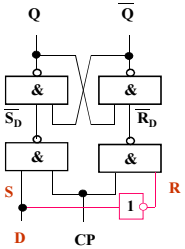


或者

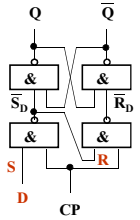




钟控D锁存器工作原理



- CP＝0，$\overline{S}_D$=$\overline{R}_D$=1，保持原态。
- 当CP＝1时，若D=0，相当于S=0，R=1，锁存器置“0”；若D=1，相当于S=1，R=0，锁存器置“1”。



- 借用输入控制电路的一个与非门对D反相后反馈到另一个与非门的输入端，作为R信号

钟控D锁存器的逻辑功能表示

- 电路功能
  - CP=0时保持

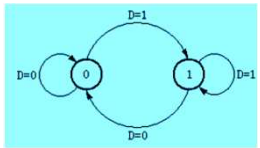
CP=1时的特性方程

$$Q^{n+1} = D$$

特性表(CP=1)

D	Q^n	Q^{n+1}	功能
0	0	0	置0
0	1	0	
1	0	1	置1
1	1	1	

状态图



$$Q^{n+1} = D\overline{Q}^n + DQ^n = D$$

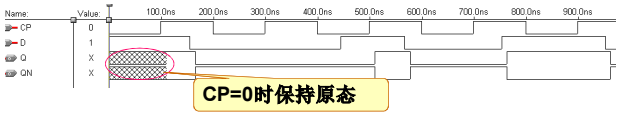
D锁存器的HDL设计

❖根据特性表，采用行为描述方式

➢D锁存器为电位触发器，假定为高电平触发

$$Q^{n+1} = D \quad (CP = 1)$$
$$Q^{n+1} = Q^{n} \quad (CP = 0)$$

```
module  D_FF_1(CP,D,Q,QN);
        input   CP,D;
        output  Q,QN;
        reg     Q,QN;
    always
      begin
        if (CP == 1)begin Q = D;QN = ~Q; end
        else        begin Q = Q;QN = QN; end
      end
endmodule
```



第三部分：时序逻辑电路设计

一． 锁存器和触发器
1. SR/D锁存器
2. D触发器
3. JK触发器

二． 有限状态机
1. Moore型有限状态机
2. Mealy型有限状态机

三． 时序逻辑电路设计分析
1. 数据寄存器
2. 移位寄存器
3. 计数器
4. Verilog HDL 设计
5. 时序电路的时序

D触发器

❖一个D触发器可以由两个反相的D锁存器构成，如下图(a)所示。图(b)为D触发器符号。

❖锁存器L1为主锁存器，L2为从锁存器

❖工作原理

➢CLK＝0；L1是通路，L2是断路，Q1←D，Q2值不变

➢CLK从0上升到1； Q2←Q1，触发时刻

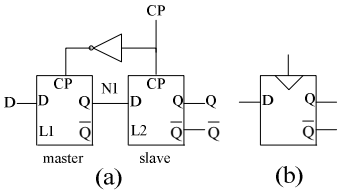
➢CLK＝1； L1是断路，L2是通路，Q1值不变， Q2←Q1

❖D触发器分类

➢主从触发器

➢边沿触发器

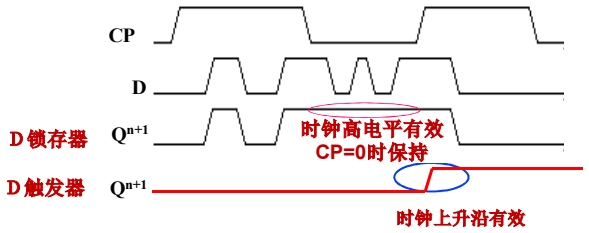
➢维持－阻塞触发器



锁存器与触发器的区别

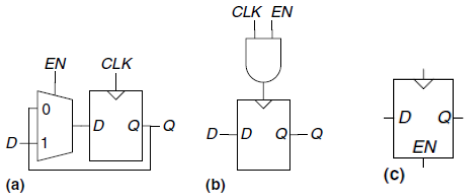
❖ 锁存器是电位（电平）触发的，只有在时钟CP有效电平（高电平CP=1或者低电平CP＝0）期间，触发器的状态才有可能发生变化。

❖ 触发器的状态变化只发生在时钟CP的有效沿（上升沿或者下降沿）期间，CP＝1、CP=0时触发器的状态不会发生变化。



带使能端的D触发器

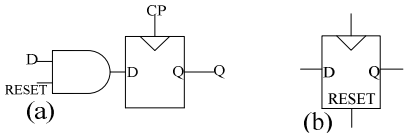
- 增加输入使能信号EN（ENable），用于确定在时钟沿是否能够载入数据，如下图所示。
  - EN＝1时，D触发器正常工作
  - EN＝0时， D触发器状态不变
  - 在时钟信号上一般不要设置逻辑，否则可能因延迟导致时序错误



(a,b)原理图，(c)电路符号

带复位功能的D触发器

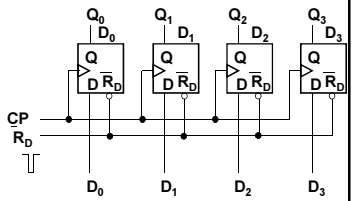
- 增加输入复位信号RESET，如下图所示。
  - 当系统加电讲触发器设置为已知状态（Q=0）
  - RESET 有效时（＝1），D触发器复位（Q=0）
  - RESET 无效时（＝0）， D触发器正常工作
- 复位方式
  - 同步复位：复位信号有效和时钟有效沿同时有效才能复位（置0）
  - 异步复位：只要复位信号有效就能复位
- 有的触发器还带有置位（SET）功能（Q=1）



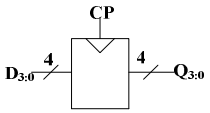
(a)原理图，(b)电路符号

由D触发器构成寄存器

- 由同一时钟控制的*N*个D触发器可以构成*N*位寄存器
  - 图(a)为4位寄存器，图(b)为电路符号
    D3D2D1D0：并行数据输入
    Q3Q2Q1Q0：并行数据输出
- 工作原理
  （1）清除（复位）
  当 $\overline{R_D}=0$， $Q_0Q_1Q_2Q_3$=0000
  （2）置数（复位端无效时）
  当CP上升沿到来是，$Q_0Q_1Q_2Q_3= D_0D_1D_2D_3$
- 工作方式（数据输入输出方式）
  - 并入并出



(a)

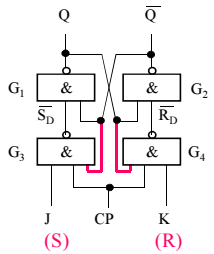


(b)

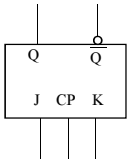
第三部分：时序逻辑电路设计

钟控JK触发器

- D触发器虽然没有约束条件，但功能较少（只有置0、置1功能）。
- JK触发器是一种功能最全面，而且没有约束条件的FF。它是在钟控RS FF的基础上，增加两条反馈线，Q反馈到R钟控门的输入端，并把R改为K；/Q反馈到S门上，并把S改名为J。



钟控JK触发器的电路结构



- 把RS＝11的无效状态变为JK触发器的翻转（计数）功能

钟控JK触发器的逻辑功能表示

- 电路功能

CP=0时为保持功能

特性表(CP=1)

J	K	Q^n	Q^{n+1}	功能
0	0	0	0	保持
0	0	1	1	
0	1	0	0	置0
0	1	1	0	
1	0	0	1	置1
1	0	1	1	
1	1	0	1	翻转
1	1	1	0	(计数)

$$Q^{n+1} = \overline{J}\overline{K}Q^n + J\overline{K}\overline{Q}^n + J\overline{K}Q^n + JK\overline{Q}^n$$
$$= (J\overline{K}\overline{Q}^n + JK\overline{Q}^n) + (\overline{J}\overline{K}Q^n + J\overline{K}Q^n)$$
$$= J\overline{Q}^n + \overline{K}Q^n$$

简化特性表(CP=1)

J	K	Q^{n+1}	功能
0	0	Q^n	保持
0	1	0	置0
1	0	1	置1
1	1	$\overline{Q}^n$	翻转

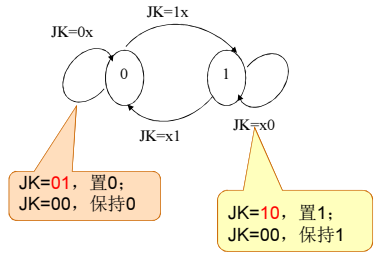
特性方程

$$Q^{n+1} = J\overline{Q}^n + \overline{K}\cdot Q^n$$

J0K0，输出不变； J0K1，输出为0；
J1K0，输出为1；J1K1，分频计数

钟控JK触发器的状态图和时序图

状态转换图



钟控JK触发器的HDL设计

- 设计分析
  - 根据CP=0和1，分2种情况，适合用if-else语句来描述
    - CP=0时保持
    - CP=1时完成JK FF的功能（根据简化特性表有4种功能，适于用case语句来描述）

简化特性表(CP=1)

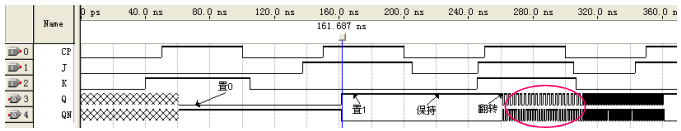
J	K	Q^{n+1}	功能
0	0	Q^n	保持
0	1	0	置0
1	0	1	置1
1	1	$\overline{Q}^n$	翻转

钟控JK触发器的源程序

采用行为描述方式

```
module   JK_FF(CP,J, K,Q,QN);
    input     CP, J, K;
    output    Q,QN;
    reg       Q,QN;
    always  @(CP or J or K)
       begin
          if (CP==0)                              //保持
           begin Q = Q;   QN = QN; end
          else if (CP==1)
              case ({J,K})
                   2'b00: begin Q = Q;      QN = QN; end   //保持
                   2'b01: begin Q = 1'b0;   QN = 1'b1; end //置0
                   2'b10: begin Q = 1'b1;   QN = 1'b0; end //置1
                   2'b11: begin Q = !Q;     QN = !QN; end //翻转
              endcase
       end
endmodule
```


钟控JK触发器的仿真波形



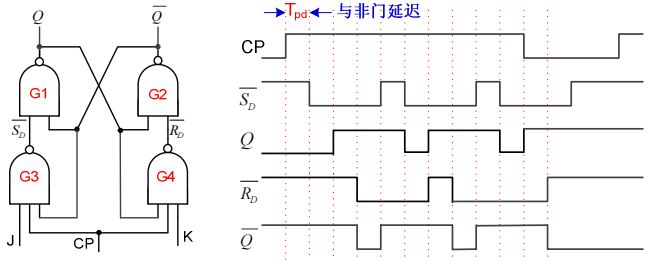
J=1、K=1时
发生空翻

❖ 钟控JK触发器比D触发器新增的功能：

当JK=11时，输出的波形翻转（可用于计数）；当JK=00时，触发器保持原来的状态。

钟控JK触发器

❖钟控JK触发器的空翻现象分析（初态Q=0，J=1，K=1）

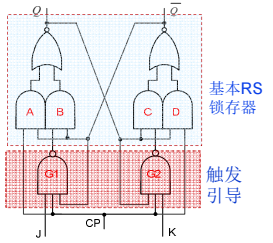


- 当CP＝1时，由于触发器初态Q＝0，门G4截止，门G3导通。2个T_{pd}后，Q端由0变为1；第3个T_{pd}后，/Q由1变为0，触发器完成第一次翻转。
- 如果CP＝1继续保持，由于/Q＝0使门G3截止，Q＝1使门G4导通，第4个T_{pd}后，/Q由0变为1，第5个T_{pd}后，Q由1变为0，又使触发器完成第二次翻转。…
- 要保证CP=1 期间JK触发器只翻转1次：3 T_{pd} < T_{CP} < 4 T_{pd} 在实际的电路中难以实现

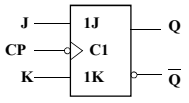
负边沿触发的JK触发器

❖电路结构

- 一个基本RS锁存器
- 一个触发引导逻辑
- 利用门电路传输延迟差异而引导触发
- G1、G2传输延迟大于A、D的翻转时间



电路结构

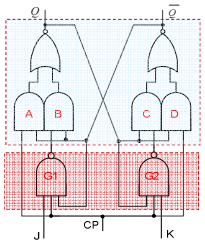


符号表示

负边沿触发的JK触发器

❖工作原理

1. CP=0时：G1、G2输出为1，门A、D封锁，故门B、C构成基本RS锁存器。/S_D=1，/R_D=1，保持功能，$Q^{n+1}=Q^n$，与J、K输入无关。
2. CP=1时：门A和D输入分别为/Q和Q，若Q^n=0，则Q^{n+1}=0；若Q^n=1，则Q^{n+1}=1，保持功能，$Q^{n+1}=Q^n$，与J、K输入无关。
3. CP↓时：
   ① 门A、D先关闭，但G1、G2还未关闭（任可视为CP=1），J、K的状态从G1、G2输出，通过门B、C进入基本RS触发器，先完成的是钟控JK触发器的功能。若JK=10，Q^{n+1}=1，置1功能。若JK=01，Q^{n+1}=0，置0功能。
   ② 之后，G1、G2关闭，输出1，/S_D=1、/R_D=1，触发器执行保持功能，即使J、K状态再发生变化也不影响触发器的状态。



电路结构

$CP\downarrow$ 时

$$Q^{n+1}=J\overline{Q^n}+\overline{K}Q^n$$

第三部分：时序逻辑电路设计

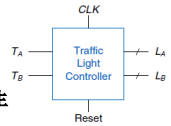
一. 锁存器和触发器
   1. SR/D锁存器
   2. D触发器
   3. JK触发器
二. 有限状态机
   1. Moore型有限状态机
   2. Mealy型有限状态机
三. 时序逻辑电路设计分析
   1. 数据寄存器
   2. 移位寄存器
   3. 计数器
   4. Verilog HDL 设计
   5. 时序电路的时序

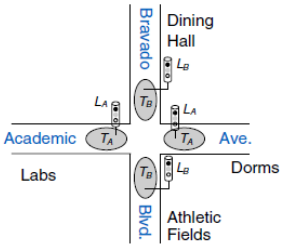
有限状态机设计实例

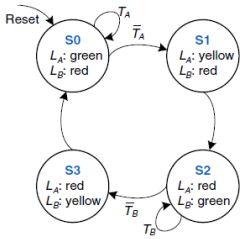
❖交通信号灯控制器

➢输入
- 传感器T_A和T_B: 为1是表示此条路上有学生；为0无学生

➢输出
- 信号灯L_A和L_B显示不同颜色



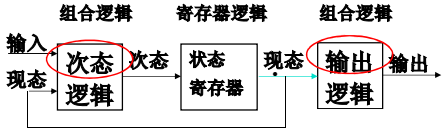




有限状态机概述

❖有限状态机（Finite State Machine，FSM）是表示有限个状态以及这些状态之间的转移和动作等行为的离散数学模型。

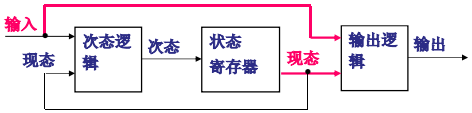
❖有限状态机是组合逻辑和寄存器逻辑的特殊组合。组合逻辑部分包括次态逻辑和输出逻辑，分别用于状态译码和产生输出信号；寄存器逻辑部分用于存储状态。



Moore型状态机典型结构

有限状态机的分类

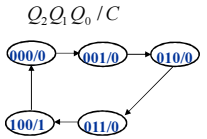
- FSM常用于时序逻辑电路设计，尤其适于设计数字系统的控制模块。具有速度快、结构简单、可靠性高、逻辑清晰、复杂问题简单化的优点。
- 根据输出信号产生的机理不同，状态机可以分成两类：
  - 摩尔(Moore)型状态机—输出信号仅与当前状态有关
  - 米里(Mealy)型状态机—输出信号与当前状态及输入信号有关



Mealy型状态机的典型结构

有限状态机的表示方法

- 状态机有2种表示方法
  - 状态图（State Diagram）、状态表（State Table）
  - 二者可以相互转换



状态（转换）图

状态（转换）表

$Q_2^nQ_1^nQ_0^n$	$Q_2^{n+1}Q_1^{n+1}Q_0^{n+1}$	C
0 0 0	0 0 1	0
0 0 1	0 1 0	0
0 1 0	0 1 1	0
0 1 1	1 0 0	0
1 0 0	0 0 0	1

有限状态机的设计方法

- 实用的状态机一般都设计为同步时序逻辑电路，它在同一个时钟信号的触发下，完成各状态之间的转移。
- 状态机设计步骤：
  1. 分析设计要求，列出全部可能状态；
  2. 画出状态转移图；
  3. 用Verilog HDL语言描述状态机，主要采用always块语句，完成3项任务：

（1）定义起始状态（敏感信号为时钟和复位信号）；
（2）用case或if-else语句描述出状态的转移（根据现态和输入产生次态）；
（3）用case或if-else语句描述状态机的输出信号（敏感信号为现态）。

状态机的设计要点

- 起始状态的选择

起始状态指电路复位后所处的状态，选择一个合理的起始状态将使整个系统简捷高效。有限状态机必须有时钟信号和复位信号。

- 状态编码方式
  - 二进制编码：采用$\log_2N$个触发器来表示这N个状态——节省逻辑资源，但可能产生毛刺
  - 格雷编码：采用$\log_2N$个触发器来表示这N个状态，同时相邻状态只有一个比特位不同。节省逻辑资源；又避免产生毛刺——在状态的顺序转换中，相邻状态每次只有一个比特位产生变化
  - 一位热码状态机编码（One-Hot State Machine Encoding）：采用N个触发器来表示这N个状态。可以避免状态机产生错误的输出，并且有时可简化输出逻辑。

对8个状态三种编码方式的对比

状态	二进制编码	格雷编码	一位热码编码
state0	000	000	00000001
state1	001	001	00000010
state2	010	011	00000100
state3	011	010	00001000
state4	100	110	00010000
state5	101	111	00100000
state6	110	101	01000000
state7	111	100	10000000

- 采用一位热码编码，虽然使用触发器较多，但可以有效节省和简化组合逻辑电路。
- FPGA有丰富的寄存器资源，门逻辑相对缺乏，采用一位热码编码可以有效提高电路的速度和可靠性，也有利于提高器件资源的利用率。

状态编码的HDL定义

- 状态编码的定义有两种方式：parameter和｀define语句

【例1】为state0, state1 ,state2 ,state3四个状态定义码字为：00，01，11，10

✓方式一：用parameter参数定义
用n个parameter常量表示n个状态

```
parameter  state0=2'b00,
           state1=2'b01,
           state2=2'b11,
           state3=2'b10;
……
  case (state)
    state0:……;
    state1:……;
    ……
```

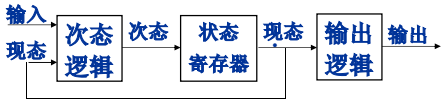
✓方式二：用｀define语句定义
用n个宏名表示n个状态

```
`define state0=2'b00 //不要加分号
`define state1=2'b01
`define state2=2'b11
`define state3=2'b10
case (state)
  `state0:……;
  `state1:……;
   ……
```

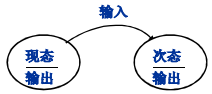

状态转换的描述

- 一般用case、casez或casex语句，比用if-else语句更清晰明了！
- 在case语句的最后，要加上default分支语句，以避免锁存器的产生。
- 状态机一般应设计为同步方式，并由一个时钟信号来触发。
- 实用的状态机都应设计为由唯一的时钟边沿触发的同步运行方式

Moore型有限状态机



Moore型状态机典型结构



Moore型状态图的表示

- Moore型状态机，其输出只为状态机当前状态的函数，而与外部输入无关。
- 外部输出是内部状态的函数。

Moore型有限状态机设计举例

【例2】设计一个序列检测器。要求检测器连续收到串行码{1101}后，输出检测标志为1，否则输出检测标志为0。

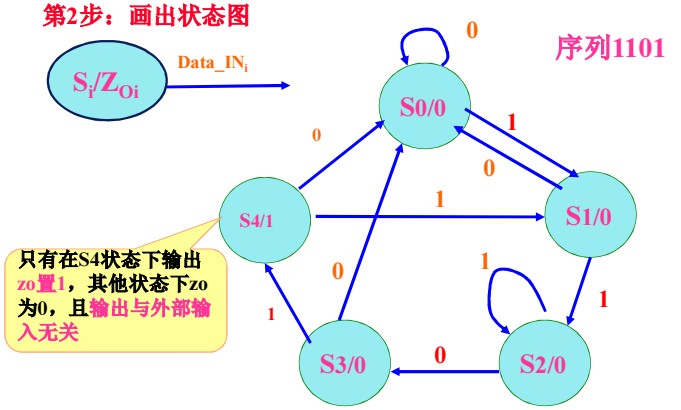
第1步：分析设计要求，列出全部可能状态：

未收到一个有效位（0）	：S0
收到一个有效位（1）	：S1
连续收到两个有效位（11）	：S2
连续收到三个有效位（110）	：S3
连续收到四个有效位（1101）	：S4

❖由于序列检测器的输出只为状态机当前状态的函数，而与外部输入无关，所以为Moore型状态机

Moore型有限状态机设计举例

第2步：画出状态图



Moore型有限状态机设计举例

第3步：用Verilog语言描述状态机

❖在程序的开头定义状态机状态的编码形式

- 用parameter或'define语句

❖复位时回到起始状态

- 敏感信号为时钟和复位信号

❖状态转换描述

- 用case或if-else语句描述出状态的转移（根据现态和输入产生次态，可与复位时回到起始状态的语句放在同一个always块中，即敏感信号为时钟和复位信号）

❖输出信号描述

- 用case语句（Mealy型状态机还要用到if-else语句）描述状态机的输出信号（单独放在一个always块中，敏感信号为现态）

序列检测器源程序（Moore型状态机）

```
module monitor(clk,clr,data,zo,state);
  parameter S0=3'b000, S1=3'b001,
  S2=3'b010,S3=3'b011,S4=3'b100; //状态编码的定义
  input clk,clr,data;
  output zo;
  output[2:0] state;                    //状态机
  reg [2:0] state;
  reg zo;
  always @(posedge clk or posedge clr)
    begin
      if (clr) state=S0; //（1）复位时回到初始状态
      else
        begin
          case (state)//  （2）状态的转移
            S0: if (data==1'b1) state=S1;
                else state=S0;
            S1: if (data==1'b1) state=S2;
                else state=S0;
            S2: if (data==1'b0) state=S3;
                else state=S2;
            S3: if (data==1'b1) state=S4;
                else state=S0;
            S4: if (data==1'b1) state=S1;
                else state=S0;
            default: state=S0;
          endcase
          zo=(state==S4)?1'b1:1'b0;  //（3）状态机的输出信号
        end
    end
endmodule
```

在S4时给zo置1——输出只为状态机当前状态的函数，而与外部输入无关

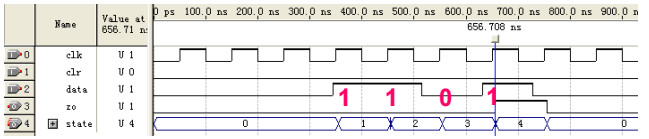
状态机的输出信号描述

- 如果输出表达式很简单，可以单独用一条赋值语句写出
- 最好将状态机的输出语句单独写在一个always块中，这样逻辑清晰，不易出错：

语句“zo=(state==S4)?1'b1:1'b0;”去掉，换成如下always块（组合逻辑）

```
always @(state)      //（3）状态机的输出信号
   begin
      case (state)
         S0:  zo=1'b0;
         S1:  zo=1'b0;
         S2:  zo=1'b0;
         S3:  zo=1'b0;
         S4:  zo=1'b1;
         default: zo=1'b0;
      endcase
   end
```


序列检测器的仿真波形

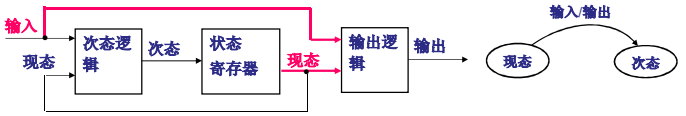


- 状态机的Verilog描述一般采用双过程描述：一个过程描述现态和次态时序逻辑；另一个过程描述输出逻辑（组合逻辑）。这样写结构清晰；将时序逻辑和组合逻辑分开描述，便于修改。
- 描述包括3个部分：
  （1）复位时回到初始状态；
  （2）状态的转移；
  （3）状态机的输出信号。

返回

第三部分：时序逻辑电路设计

Mealy型有限状态机

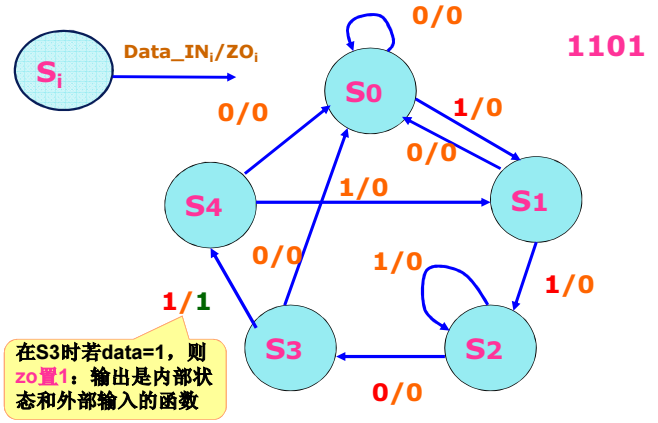


Mealy型状态机的典型结构　　Mealy型状态图的表示

- Mealy型状态机，其输出不仅与状态机当前状态有关，而且与输入有关——外部输出是内部状态和外部输入的函数。
- 在“Mealy型状态图的表示”中，每个圆圈表示状态机的一个状态，每个箭头表示状态之间的一次转移；引起状态转换的输入信号及产生的输出信号标注在箭头上。

【例3】将【例2】采用Mealy型状态机实现。

序列检测器的Mealy型有限状态机状态图



序列检测器源程序（Mealy型状态机）

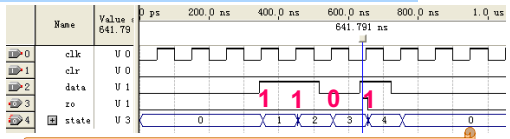
```
module monitor2_good(clk,clr,data,zo,state);
  parameter S0=3'b000, S1=3'b001,
  S2=3'b010,S3=3'b011,S4=3'b100;
  input clk,clr,data;
  output zo;
  output[2:0] state;
  reg [2:0] state;
  reg zo;
  always @(posedge clk or posedge clr)
    begin
      if (clr) state=S0; //（1）复位时回到初始状态
      else
        begin
          case (state) //（2）状态的转移
            S0:  if (data==1'b1) state=S1;
                 else state=S0;
            S1:  if (data==1'b1) state=S2;
                 else state=S0;
            S2:  if (data==1'b0) state=S3;
                 else state=S2;
            S3:  if (data==1'b1) state=S4;
                 else state=S0;
            S4:  if (data==1'b1) state=S1;
                 else state=S0;
            default: state=S0;
          endcase
        end
    end
endmodule
```


序列检测器源程序及仿真波形

```
always @(state)     //（3）状态机的输出信号
  begin
    case (state)
      S0:  zo=1'b0;
      S1:  zo=1'b0;
      S2:  zo=1'b0;
      S3:  if(data==1'b1) zo=1'b1;
           else zo=1'b0;
      S4:  zo=1'b0;
      default: zo=1'b0;
    endcase
  end
endmodule
```

与Moore型比较

在S3时若data=1，则zo置1：输出是内部状态和外部输入的函数



❖采用Mealy型状态机描述序列检测器与采用Moore型状态机的仿真波形稍有区别！

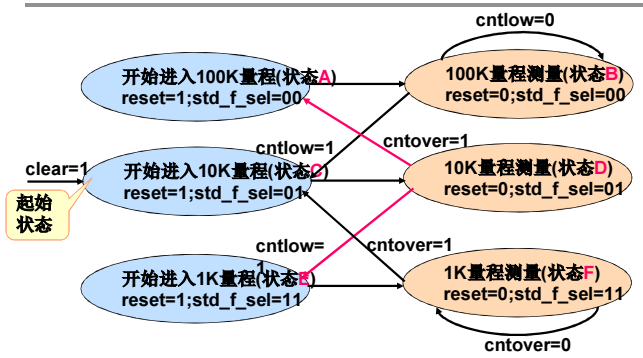
波形比较

自动转换量程频率计控制器

【例4】设计一个自动转换量程频率计控制器，要求根据超量程或欠量程输入信号，自动切换到合适的量程；并输出复位频率计计数器的信号，以及选择标准时基的信号。

- ❖假定频率计有3个量程：1K、10K、100K
- ❖控制器有6个工作状态：进入100K量程、100K量程测量、进入10K量程、10K量程测量、进入1K量程、1K量程测量。
- ❖输入信号
  - ➢clk：系统时钟；
  - ➢clear：系统复位信号；
  - ➢cntover：超量程；
  - ➢cntlow：欠量程。
- ❖输出信号
  - ➢reset：转换到某量程时复位频率计计数器信号；
  - ➢std_f_sel：选择标准时基信号

频率计控制器状态转移图（ Moore型状态机）



- 起始状态选为一个中间量程，为C（进入10K量程）。
- 从“进入某档量程”转移到“该档量程测量”为无条件转移；而当在测量时，根据超量程或欠量程，切换到适宜的量程。

频率计控制器源程序（1/3）

```
module frequency_right(clk, clear, cntover, cntlow, reset, std_f_sel,state);
    input clk, clear, cntover, cntlow;
    output reset;              //量程转换开始时复位频率计计数器
    output[1:0] std_f_sel;  //选择标准时基
    output[5:0] state;
    reg [5:0] state;
    reg reset;
    reg[1:0] std_f_sel;
    parameter start_f100k=6'b000001, f100k_cnt=6'b000010,
                start_f10k=6'b000100, f10k_cnt=6'b001000,
                start_f1k=6'b010000, f1k_cnt=6'b100000;
// ① 复位时回到起始状态
 always@(posedge clk or posedge clear)
    if(clear) state=start_f10k;//复位时“进入10K量程”为起始状态
    else
```

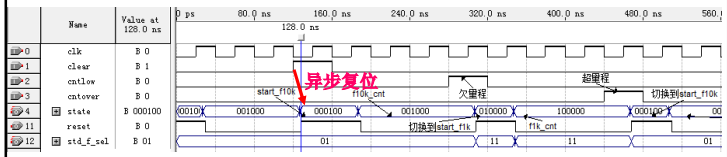

频率计控制器源程序（2/3）

```
// ② 状态的转换（根据现态和输入产生次态）
    begin
      case(state)
        start_f100k: state=f100k_cnt;
        f100k_cnt:   if (cntlow) state= start_f10k;    //欠量程
                     else state= f100k_cnt;
        start_f10k:  state=f10k_cnt;
        f10k_cnt:    if (cntover) state= start_f100k;  //超量程
                     else if (cntlow) state= start_f1k; //欠量程
        start_f1k:   state=f1k_cnt;
        f1k_cnt:     if (cntover) state= start_f10k;   //超量程
                     else state= f1k_cnt;
        default: state=start_f10k;       //默认状态为起始状态“进入10K量程”
      endcase
    end
```


频率计控制器源程序（3/3）

```
// ⑧ 状态机的输出信号
 always@(state)
    begin
      case(state)
        start_f100k:  begin reset=1; std_f_sel=2'b00; end
        f100k_cnt:    begin reset=0; std_f_sel=2'b00; end
        start_f10k:   begin reset=1; std_f_sel=2'b01; end
        f10k_cnt:     begin reset=0; std_f_sel=2'b01; end
        start_f1k:    begin reset=1; std_f_sel=2'b11; end
        f1k_cnt:      begin reset=0; std_f_sel=2'b11; end
        default:      begin reset=1; std_f_sel=2'b01; end
      endcase
   end
endmodule
```

频率计控制器时序仿真波形



第三部分：时序逻辑电路设计

寄存器的分类

- 寄存器
  - 计算机中重要部件，用于存放一组二进制代码，如指令、参加运算的数据、运算结果等，广泛用于各类数字系统中；
  - 由触发器组成，一个触发器能储存1位二进制代码。还包括接收数据的控制门电路，以便在同一个接收命令作用下使各触发器同时接收数据；
  - 触发器的触发方式决定了寄存器的触发方式。数据寄存器常用的是上升沿触发的D型触发器（边沿触发）和电位触发器，较少采用主-从触发器。
  - 寄存器的操作：读/写/复位（清零）

寄存器
- 数据寄存器
  - 数据寄存器（多位D型触发器组成）
  - 数据锁存器（多位电位触发器组成）
  - 寄存器阵列
- 移位寄存器
  - 单向移位寄存器
  - 双向移位寄存器

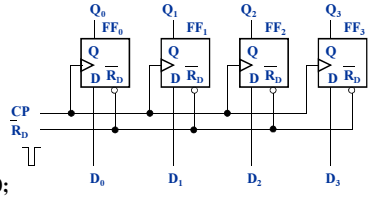
数据寄存器

- 数据寄存器具有接收、存放和传输数据的功能，是由多位边沿触发器组成的用于保存一组二进制代码的寄存单元。
- 各种类型的触发器都具有置0、置1和保持（记忆）功能，它们都可以用来构成寄存器，而用D触发器构成寄存器最为方便。

- 4位D型寄存器电路结构（N=4）

$D_3D_2D_1D_0$：并行数据输入

$Q_3Q_2Q_1Q_0$：并行数据输出



- 工作原理

（1）清除（复位）

当 $\overline{R_D}=0$ ， $Q_0Q_1Q_2Q_3$=0000;

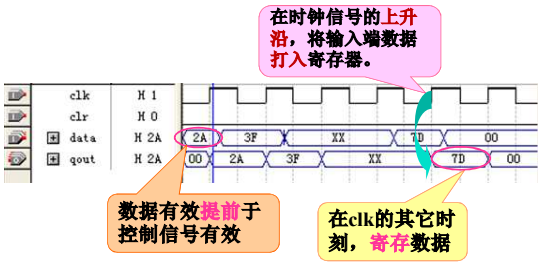
（2）置数（复位端无效时）

当 $CP=CP\uparrow$ ， $Q_0Q_1Q_2Q_3=D_0D_1D_2D_3$;

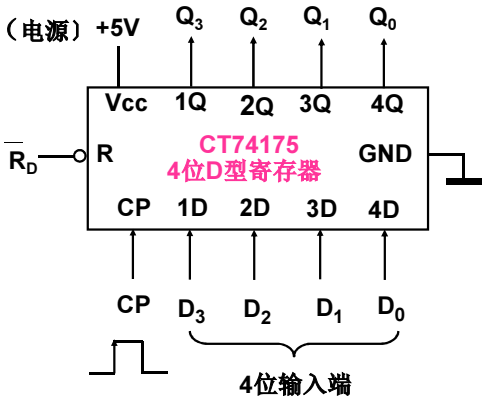
- 工作方式（数据输入输出方式）——只能并入并出

数据寄存器

❖当时钟信号的上升沿或下降沿到来时，将输入端数据打入寄存器，即此时输出信号等于输入信号；在时钟信号的其它时刻，输出端保持刚才输入的数据，即为寄存状态，而不管此时输入信号是否变化。

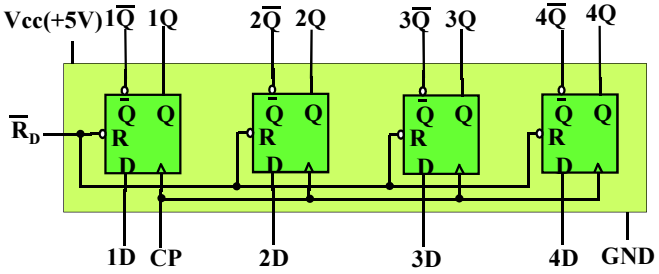


4位二进制数寄存器CT74175



CT74175电路结构

❖由边沿触发器组成
❖4级D 触发器的输入构成4位输入端1D、2D、3D、4D，1Q、2Q、3Q、4Q构成4位输出端
❖各触发器的时钟连接在一起，作为整个寄存器的时钟端CP
❖异步置0端连在一起，作为整个寄存器的复位端



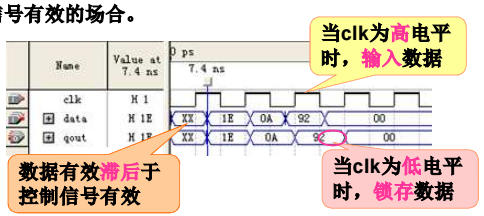
8位数据寄存器的HDL设计

【例】用always块语句描述的8位数据寄存器

```
module reg_8bit(qout,data,clk,clr);
    output[7:0] qout;
    input [7:0] data;
    input clk,clr;
    reg [7:0] qout;
    always @(posedge clk or posedge clr)    //沿触发
      begin
        if(clr) qout=0;                     //异步清零
        else    qout= data;
      end
endmodule
```

数据锁存器

- **数据锁存器**：由多位电位触发器组成的用于保存一组二进制代码的寄存单元。
- 功能：当输入控制信号（如时钟）为高电平时，门是打开的，输出信号等于输入信号；当输入控制信号为低电平时，门是关闭的，输出端保持刚才输入的数据，即为锁存状态，而不管此时输入信号是否变化。
- 通常由电平信号来控制，属于电平敏感型，适于数据有效滞后于控制信号有效的场合。



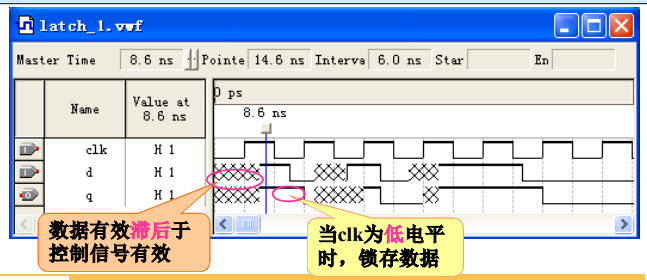
数据寄存器与数据锁存器的区别

- 数据寄存器和数据锁存器的区别：
  - 数据寄存器由边沿触发的触发器组成。通常由同步时钟信号来控制，属于脉冲敏感型，适于数据有效提前于控制信号（一般为时钟信号）有效、并要求同步操作的场合。
  - 数据锁存器由电位触发器（即D锁存器）组成。一般由电平信号来控制，属于电平敏感型，适于数据有效滞后于控制信号有效的场合。

1位数据锁存器的HDL设计

【例】1位数据锁存器

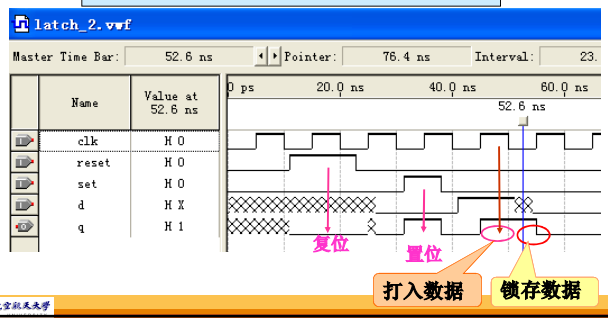
```
module latch_1(q,d,clk);
    output q;
    input d,clk ;
    assign q=clk?d:q; /*当时钟信号为高电平时，将输入端信号打入锁存器；
                        当时钟信号为低电平时，锁存原来已打入的数据。*/
endmodule
```



带置位和复位端的1位数据锁存器的HDL设计

【例】带置位和复位端的1位数据锁存器

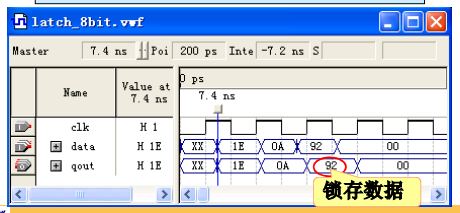
```
module latch_2(q,d,clk,set,reset);
    output q;
    input d,clk ,set,reset;
    assign q= reset ? 0: (set ? 1:(clk ? d:q));
endmodule
```



8位数据锁存器的HDL设计

【例】用always块语句描述的8位数据锁存器

```
module latch_8bit(qout,data,clk);
    output[7:0] qout;
    input [7:0] data;
    input clk;
    reg [7:0] qout;
    always @(clk or data)    //电平敏感
      if(clk) qout=data;
endmodule
```



第三部分：时序逻辑电路设计

移位寄存器

- 在计算机中，常要求寄存器有“移位”功能。例如，移位相加乘法器进行乘法运算时，要求将乘数右移，被乘数左移；除法运算时，要求将余数左移；将并行传递的数转换成串行数据以及将串行传递的数转换成并行数据的过程中，需要移位。
- 具有移位功能的寄存器称为移位寄存器，每来一个时钟脉冲，寄存器中数据就依次向左或向右移一位。
- 分类
  - 左移移位寄存器，右移移位寄存器，双向移位寄存器
- 数据输入方式
  - 串行输入，并行输入
- 数据输出方式
  - 串行输出：右移寄存器、左移寄存器
  - 并行输出：全部触发器的输出作为电路的输出
- 根据数据输入/输出方式，移位寄存器的工作方式有
  - 串入串出、串入并出、并入串出、并入并出

4位右移移位寄存器

- 电路结构（N=4右移）

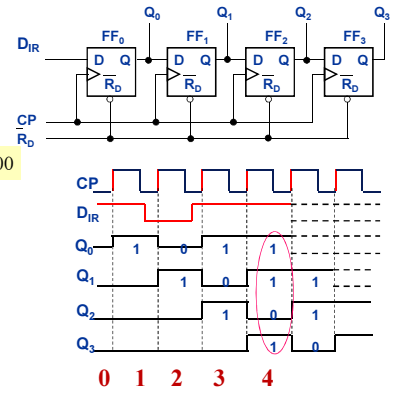
  D_{IR}：右移串行数据输入1011

- 工作原理
  - 复位：

$$\overline{R_D}=0 \rightarrow Q_3Q_2Q_1Q_0=0000$$

  - 移位：

$$Q_0^{n+1}=D_0\mathrm{CP}\uparrow=D_{IR}\mathrm{CP}\uparrow$$
$$Q_1^{n+1}=D_1\mathrm{CP}\uparrow=Q_0^{n}\mathrm{CP}\uparrow$$
$$Q_2^{n+1}=D_2\mathrm{CP}\uparrow=Q_1^{n}\mathrm{CP}\uparrow$$
$$Q_3^{n+1}=D_3\mathrm{CP}\uparrow=Q_2^{n}\mathrm{CP}\uparrow$$



4位右移移位寄存器工作方式

❖工作方式

➢串入并出

串并转换（需要N个CP周期），经过4个CP，串行输入的4位数据全部移入移位寄存器中，并从$Q_3Q_2Q_1Q_0$并行输出1011

➢串入串出

把最右边的触发器的输出作为电路的输出。经过4个CP后， Q3 输出的是最先串行输入的数据。从每个触发器Q端输出的波形相同，但后级触发器Q端输出波形比前级触发器Q端输出波形滞后一个时钟周期。 把工作于串入串出方式的移位寄存器称为“延迟线”（第N级FF延迟N个CP周期）

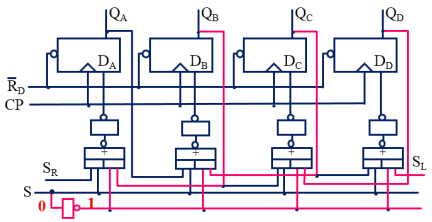
4位串行输入、串/并行输出双向移位寄存器

❖S=0，左移：数据从SL端串行输入，顺序左移，经过4个CP，串行输入的4位数据全部移入移位寄存器中，可从输出端并行输出；也继续左移，并通过QA端实现串行输出。

❖S=1，右移：数据从SR端串行输入，也可实现并行或串行输出。

❖CP：时钟信号

❖/RD：复位信号，低电平有效



$$D_A = S_R S + Q_B \overline{S}$$
$$D_B = Q_A S + Q_C \overline{S}$$
$$D_C = Q_B S + Q_D \overline{S}$$
$$D_D = Q_C S + S_L \overline{S}$$

4位串行输入、串/并行输出双向移位寄存器工作原理

❖复位 $\overline{R_D} = 0 \rightarrow Q_A Q_B Q_C Q_D = 0000$

❖移位

（1）当S=0 时：左移移位寄存器

串入并出——数据从S_L端串行输入，顺序左移，$D_A=Q_B$， $D_B=Q_C$，$D_C=Q_D$， $D_D=S_L$。经过4个CP，串行输入的4位数据全部移入移位寄存器中，并从$Q_AQ_BQ_CQ_D$并行输出。

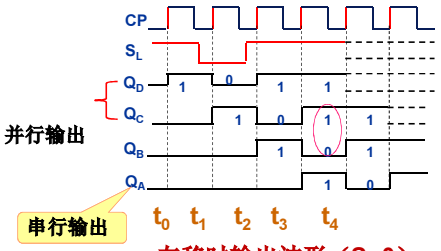
串入串出——左移（右移）移位寄存器把最左边（右边）的触发器的输出作为电路的输出。从每个触发器Q端输出的波形相同，但后级触发器Q端输出波形比前级触发器Q端输出波形滞后一个时钟周期。左移时，经过4个CP，最先串行输入的1位数据从Q_A输出，下一个CP上升沿到来时，Q_A输出串行移入的第2个数据……。

（2）S＝1时：右移移位寄存器

数据从S_R端串行输入，顺序右移，$D_A=S_R$， $D_B=Q_A$， $D_C=Q_B$， $D_D=Q_C$。

4位串行输入、串/并行输出双向移位寄存器输出波形

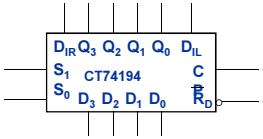
输入数据：1011



左移时输出波形（S=0）

4位双向移位寄存器CT74194

S_1S_0：功能控制输入端
$D_3D_2D_1D_0$：并行数据输入端
$Q_3Q_2Q_1Q_0$：数据输出
D_{IR}：右移串行输入（Q_3为串行输出端）
D_{IL}：左移串行输入（Q_0为串行输出端）



功能表

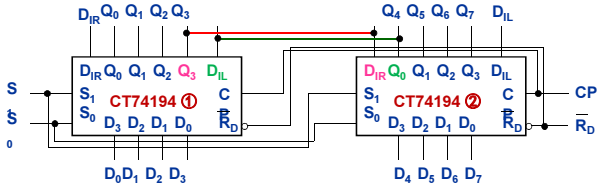
$\overline{R}_D$	CP	S_1	S_0	功能
0	X	X	X	置零
1	↑	0	0	保持
1	↑	0	1	右移
1	↑	1	0	左移
1	↑	1	1	并行输入

工作方式

- 串入并出——串并转换
- 串入串出——延迟线
- 并入串出——并串转换
- 并入并出——数据预置

移位寄存器扩展方法

用2片4位移位寄存器扩展为8位移位寄存器



- 将片①的Q_3接至片②的D_{IR}，当S1S0=01时，右移，片②的输出Q_3作为整个电路的右移串行输出端（Q_7）。
- 将片②的Q_0接至片① 的D_{IL}，当S1S0=10时，左移，片①的输出Q_0 作为整个电路的左移串行输出端Q_0。
- 同时把两片的S_1、S_0、CP和/R_D分别并联。

集成移位寄存器的用途

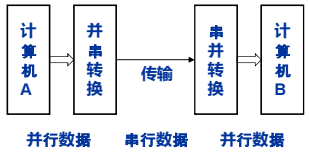
主要用途

- 数据保存与移位
- 计算机串行通信中的并串转换及串并转换

  串行通信——数据在一根传输线上一位一位地顺序传送。

  并行通信——数据以字节（字）为单位在多根传输线上同时传送。
- 移存型计数器（环形计数器，扭环形计数器）

  利用移位寄存器组成的计数器叫做移存型计数器。



第三部分：时序逻辑电路设计

计数器

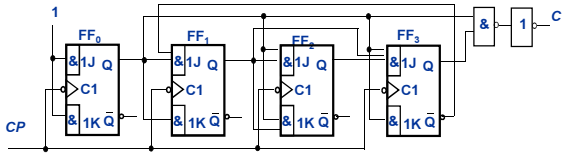
- 同步计数器
- 异步计数器
- 集成计数器

同步计数器

- 计数器是可以统计输入脉冲个数的器件
- 计数器的用途
  - （脉冲）计数
  - 计时
  - 定时（定时器）
  - 分频
  - 产生节拍脉冲（顺序脉冲）和序列脉冲
- 计数器的分类
  - 时钟方式：根据计数器中触发器时钟端的连接方式，分为同步计数器，异步计数器
  - 计数方式：二进制计数器，十进制计数器，M进制计数器
  - 状态变化：根据计数器中的状态变化规律分为加法计数器，减法计数器，加/减法计数器

同步计数器的分析方法

【例】分析下图电路，说明电路的特点。



解：(1)写出逻辑表达式（4个负边沿触发的JK触发器组成的电路）

$J_0 = K_0 = 1;$

$J_1 = \overline{Q}_3^n Q_0^n, K_1 = Q_0^n;$

$J_2 = K_2 = Q_1^n Q_0^n;$

$J_3 = Q_2^n Q_1^n Q_0^n, K_3 = Q_0^n$

$C = \overline{\overline{Q_3^n Q_0^n}} = Q_3^n Q_0^n$

$Q_0^{n+1} = J_0\overline{Q}_0^n + \overline{K}_0 Q_0^n = \overline{Q}_0^n$

$Q_1^{n+1} = J_1\overline{Q}_1^n + \overline{K}_1 Q_1^n = \overline{Q}_3^n Q_0^n \overline{Q}_1^n + \overline{Q}_0^n Q_1^n$

$Q_2^{n+1} = J_2\overline{Q}_2^n + \overline{K}_2 Q_2^n = Q_1^n Q_0^n \overline{Q}_2^n + \overline{Q_1^n Q_0^n} Q_2^n$

$Q_3^{n+1} = J_3\overline{Q}_3^n + \overline{K}_3 Q_3^n = Q_2^n Q_1^n Q_0^n \overline{Q}_3^n + \overline{Q}_0^n Q_3^n$

$CP_0 = CP_1 = CP_2 = CP_3 = CP \downarrow$ —— 同步电路

状态转换表

（2）计算并列出状态转换表

$Q_3^n Q_2^n Q_1^n Q_0^n$	$Q_3^{n+1} Q_2^{n+1} Q_1^{n+1} Q_0^{n+1}$	C
0 0 0 0	0 0 0 1	0
0 0 0 1	0 0 1 0	0
0 0 1 0	0 0 1 1	0
0 0 1 1	0 1 0 0	0
0 1 0 0	0 1 0 1	0
0 1 0 1	0 1 1 0	0
0 1 1 0	0 1 1 1	0
0 1 1 1	1 0 0 0	0
1 0 0 0	1 0 0 1	0
1 0 0 1	0 0 0 0	1
1 0 1 0	1 0 1 1	0
1 0 1 1	0 1 0 0	1
1 1 0 0	1 1 0 1	0
1 1 0 1	0 1 0 0	1
1 1 1 0	1 1 1 1	0
1 1 1 1	0 0 0 0	1

$Q_0^{n+1} = \overline{Q}_0^n$

$Q_1^{n+1} = \overline{Q}_3^n Q_0^n \overline{Q}_1^n + \overline{Q}_0^n Q_1^n$

$Q_2^{n+1} = Q_1^n Q_0^n \overline{Q}_2^n + \overline{Q_1^n Q_0^n} Q_2^n$

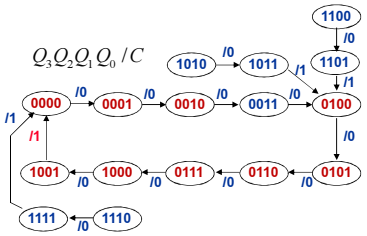
$Q_3^{n+1} = Q_2^n Q_1^n Q_0^n \overline{Q}_3^n + \overline{Q}_0^n Q_3^n$

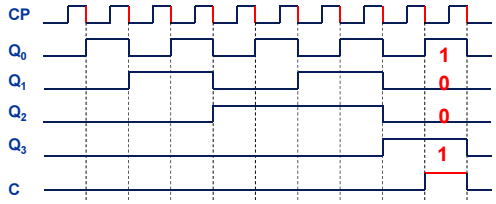
$C = Q_3^n Q_0^n$

状态转换图和时序图

（3）画状态转换图（或时序图）

- 画状态转换图时一定要画出全部状态的变化。
- 画时序图时只画出有效状态构成的计数循环的变化；注意触发器的时钟特性！





CP下降沿时触发器翻转！

当$Q_3Q_2Q_1Q_0$=1001时，C=1；下一个CP下降沿到来时，$Q_3Q_2Q_1Q_0$变为0000，完成一个计数循环。

电路功能

（4）电路功能说明

根据状态转换图，可知为同步十进制加法计数器也称为同步二进制（模10）加法计数器），并且有自启动能力

- 计数器——由若干状态构成一个计数循环
- 同步——构成电路的全部FF的时钟端连接在一起
- 十进制——计数循环的状态个数为10（模10计数器）
- 加法——计数状态按递增方向变化
- 自启动——不存在死循环，计数循环以外的状态，都能回到计数循环中来
- 死循环（无效循环）——由无效状态构成的循环.

同步计数器的特点

- 所有触发器的时钟端并联在一起，作为计数器的时钟端
- 各触发器同时翻转，不存在时钟到各触发器输出的传输延迟的积累
- 由于其工作频率只与一个触发器的时钟到输出的传输延迟有关，所以它的工作频率比异步计数器高。
- 由于计数器各触发器几乎是同时翻转的，因此，各触发器输出波形的偏移为各触发器时钟到输出的延迟之差，同步计数器输出经译码后所产生的尖峰信号宽度比较小。
- 缺点：结构比较复杂（各触发器的输入由多个Q输出相与得到），所用元件较多。

设计计数器，不能存在死循环（无效循环）

同步计数器

【例】设计一个同步十六进制数加法计数器，按照0→1→2→…→F→0的方式循环计数，当计数到F后下一个计数时钟到达时产生进位输出1。

解：

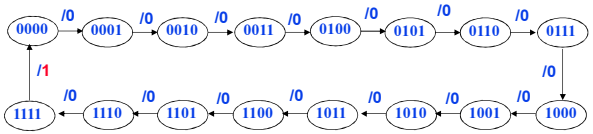
（1）实际上是一个同步4位二进制加法计数器（模16）

采用4个负沿触发JK触发器

CP为时钟信号，CP下跳沿时计数

C为进位输出

（2）根据题意，计数器的状态转换图如下。



同步计数器

解（续1）：

（3）根据状态图写状态表，也可直接写出状态表。

原态 $Q_3^n\ Q_2^n Q_1^n Q_0^n$	次态 $Q_3^{n+1}\ Q_2^{n+1} Q_1^{n+1} Q_0^{n+1}$	C
0 0 0 0	0 0 0 1	0
0 0 0 1	0 0 1 0	0
0 0 1 0	0 0 1 1	0
0 0 1 1	0 1 0 0	0
0 1 0 0	0 1 0 1	0
0 1 0 1	0 1 1 0	0
0 1 1 0	0 1 1 1	0
0 1 1 1	1 0 0 0	0
1 0 0 0	1 0 0 1	0
1 0 0 1	1 0 1 0	0
1 0 1 0	1 0 1 1	0
1 0 1 1	1 1 0 0	0
1 1 0 0	1 1 0 1	0
1 1 0 1	1 1 1 0	0
1 1 1 0	0 0 0 1	0
1 1 1 1	0 0 0 0	1

同步计数器

解（续2）：（4）根据状态表写逻辑表达式（JK触发器的特性方程）

$$
\begin{aligned}
Q_0^{n+1} &= \overline{Q_3^n}\overline{Q_2^n}\overline{Q_1^n}\overline{Q_0^n} + \overline{Q_3^n}\overline{Q_2^n}\overline{Q_1^n}Q_0^n + \overline{Q_3^n}\overline{Q_2^n}Q_1^n\overline{Q_0^n} + \overline{Q_3^n}\overline{Q_2^n}Q_1^nQ_0^n \\
&\quad + Q_3^n\overline{Q_2^n}\overline{Q_1^n}\overline{Q_0^n} + Q_3^n\overline{Q_2^n}\overline{Q_1^n}Q_0^n + Q_3^n\overline{Q_2^n}Q_1^n\overline{Q_0^n} + Q_3^nQ_2^nQ_1^n\overline{Q_0^n} \\
&= \overline{Q_0^n} \\
Q_1^{n+1} &= \overline{Q_3^n}\overline{Q_2^n}\overline{Q_1^n}Q_0^n + \overline{Q_3^n}\overline{Q_2^n}Q_1^n\overline{Q_0^n} + \overline{Q_3^n}Q_2^n\overline{Q_1^n}Q_0^n + \overline{Q_3^n}Q_2^nQ_1^n\overline{Q_0^n} \\
&\quad + Q_3^n\overline{Q_2^n}\overline{Q_1^n}Q_0^n + Q_3^n\overline{Q_2^n}Q_1^n\overline{Q_0^n} + Q_3^nQ_2^n\overline{Q_1^n}Q_0^n + Q_3^nQ_2^nQ_1^n\overline{Q_0^n} \\
&= Q_0^n\overline{Q_1^n} + \overline{Q_0^n}Q_1^n \\
Q_2^{n+1} &= \overline{Q_3^n}\overline{Q_2^n}Q_1^nQ_0^n + \overline{Q_3^n}Q_2^n\overline{Q_1^n}\overline{Q_0^n} + \overline{Q_3^n}Q_2^n\overline{Q_1^n}Q_0^n + \overline{Q_3^n}Q_2^nQ_1^n\overline{Q_0^n} \\
&\quad + Q_3^n\overline{Q_2^n}\overline{Q_1^n}Q_0^n + Q_3^n\overline{Q_2^n}Q_1^n\overline{Q_0^n} + Q_3^nQ_2^n\overline{Q_1^n}Q_0^n + Q_3^nQ_2^nQ_1^n\overline{Q_0^n} \\
&= Q_0^nQ_1^n\overline{Q_2^n} + \overline{Q_0^n}Q_1^nQ_2^n \\
Q_3^{n+1} &= \overline{Q_3^n}Q_2^nQ_1^nQ_0^n + Q_3^n\overline{Q_2^n}\overline{Q_1^n}\overline{Q_0^n} + Q_3^n\overline{Q_2^n}\overline{Q_1^n}Q_0^n + Q_3^n\overline{Q_2^n}Q_1^n\overline{Q_0^n} \\
&\quad + Q_3^nQ_2^n\overline{Q_1^n}\overline{Q_0^n} + Q_3^nQ_2^n\overline{Q_1^n}Q_0^n + Q_3^nQ_2^nQ_1^n\overline{Q_0^n} \\
&= Q_0^nQ_1^nQ_2^n\overline{Q_3^n} + \overline{Q_0^n}Q_1^nQ_2^nQ_3^n
\end{aligned}
$$

同步计数器

解（续3）：

（5）根据特性方程反推出4个JK触发器的J、K端驱动方程。

因为：JK触发器的通用特性方程是：

$$Q^{n+1} = J\overline{Q^n} + \overline{K}Q^n$$

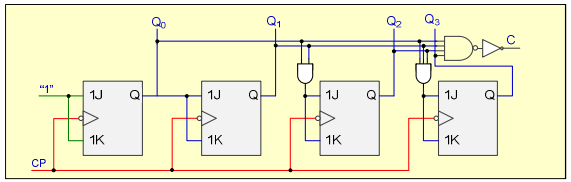
所以：4个触发器的驱动方程分别是：

$$
\begin{aligned}
J_0 &= K_0 = 1 \\
J_1 &= K_1 = Q_0^n \\
J_2 &= K_2 = Q_0^nQ_1^n \\
J_3 &= K_3 = Q_0^nQ_1^nQ_2^n
\end{aligned}
$$

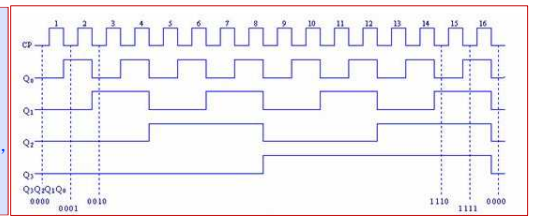
$$
\begin{aligned}
Q_0^{n+1} &= \overline{Q_0^n} \\
Q_1^{n+1} &= Q_0^n\overline{Q_1^n} + \overline{Q_0^n}Q_1^n \\
Q_2^{n+1} &= Q_0^nQ_1^n\overline{Q_2^n} + \overline{Q_0^nQ_1^n}Q_2^n \\
Q_3^{n+1} &= Q_0^nQ_1^nQ_2^n\overline{Q_3^n} + \overline{Q_0^nQ_1^nQ_2^n}Q_3^n \\
C &= Q_0^nQ_1^nQ_2^nQ_3^n
\end{aligned}
$$

同步计数器

解（续4）：（6）画出逻辑图。



Q_0、Q_1、Q_2、Q_3的周期分别是CP周期的2、4、8、16倍，实际上是分别对CP进行了2、4、8、16分频，因此二进制计数器也称为分频器。



二进制加法计数器的特点

- 二进制加法计数器的特点
  - 计数器中触发器的状态按照二进制数的规律变化，构成计数循环的状态个数（模值）为2^n（n是触发器的级数）。
  - 二进制计数器没有非编码状态，所以不存在不能自启动的问题。
  - Q_0、Q_1、Q_2、Q_3的周期分别是计数脉冲（CP）周期的2倍、4倍、8倍、16倍，也就是说Q_0、Q_1、Q_2、Q_3分别对CP波形进行了2分频、4分频、8分频、16分频，因此二进制计数器也称为分频器。

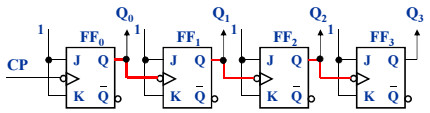
异步计数器

- 异步计数器也有二进制、十进制、任意进制等类型
- 异步计数器的特点
  - 输入系统时钟脉冲只作用于最低位触发器，高位触发器的时钟信号往往是由低一位触发器的输出提供的，高位触发器的翻转有待低一位触发器翻转后才能进行。
  - 由于每一级触发器都存在传输延迟，因此计数器工作速度慢，而且，位数越多计数越慢。在大型数字设备中较少采用。
  - 对计数器状态进行译码时，由于触发器不同步，译码器输出会出现尖峰脉冲（位数越多，尖峰信号也就越宽），使仪器设备产生误动作。
  - 优点：结构比较简单，所用元件较少。

异步二进制计数器

- 电路结构特点
  - 全部由JK触发器构成
  - 第一级FF的CP由系统时钟控制，其余各级FF的CP端由前级FF的Q端或/Q端控制

【例】分析异步二进制（M=16）加法计数器电路（N=4）



FF_1、FF_2、FF_3的CP端分别接前级触发器的Q端

（1）状态方程

$$Q_0^{n+1}=\overline{Q_0^n}\cdot CP\downarrow;\ Q_1^{n+1}=\overline{Q_1^n}\cdot Q_0\downarrow;$$
$$Q_2^{n+1}=\overline{Q_2^n}\cdot Q_1\downarrow;\ Q_3^{n+1}=\overline{Q_3^n}\cdot Q_2\downarrow;$$

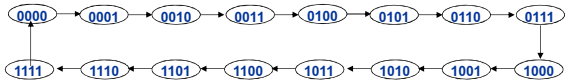
异步二进制计数器的状态转换表

（2）列出状态转换表

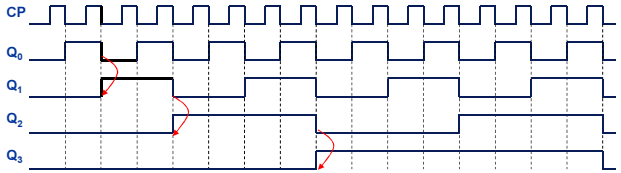
Q_3^n Q_2^n Q_1^n Q_0^n	Q_3^{n+1} Q_2^{n+1} Q_1^{n+1} Q_0^{n+1}
0 0 0 0	0 0 0 1
0 0 0 1	0 0 1 0
0 0 1 0	0 0 1 1
0 0 1 1	0 1 0 0
0 1 0 0	0 1 0 1
0 1 0 1	0 1 1 0
0 1 1 0	0 1 1 1
0 1 1 1	1 0 0 0
1 0 0 0	1 0 0 1
1 0 0 1	1 0 1 0
1 0 1 0	1 0 1 1
1 0 1 1	1 1 0 0
1 1 0 0	1 1 0 1
1 1 0 1	1 1 1 0
1 1 1 0	1 1 1 1
1 1 1 1	0 0 0 0

异步二进制计数器的状态转换图和时序图

（3）状态转换图 $Q_3Q_2Q_1Q_0$

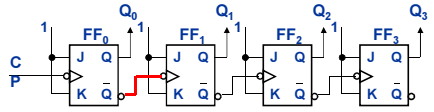


时序图



异步二进制减法计数器

【例】分析下图电路



FF_1、FF_2、FF_3的CP端分别接前级触发器的/Q端

（1）状态方程

$$Q_0^{n+1}\quad Q_0^n\cdot CP\downarrow;Q_1^{n+1}\quad \overline{Q_1^n}\cdot\overline{Q_0}\downarrow\quad \overline{Q_1^n}\cdot Q_0\uparrow;$$

$$Q_2^{n+1}=\overline{Q_2^n}\cdot\overline{Q_1}\downarrow=\overline{Q_2^n}\cdot Q_1\uparrow;Q_3^{n+1}=\overline{Q_3^n}\cdot\overline{Q_2}\downarrow=\overline{Q_3^n}\cdot Q_2\uparrow;$$

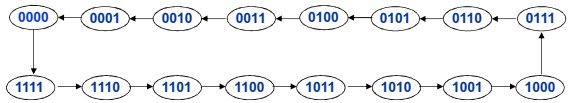
异步二进制减法计数器的状态转换表

（2）列出状态转换表

Q_3^n	Q_2^n	Q_1^n	Q_0^n	Q_3^{n+1}	Q_2^{n+1}	Q_1^{n+1}	Q_0^{n+1}
0	0	0	0	1	1	1	1
0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	1
0	0	1	1	0	0	1	0
0	1	0	0	0	0	1	1
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	1
0	1	1	1	0	1	1	0
1	0	0	0	0	1	1	1
1	0	0	1	1	0	0	0
1	0	1	0	1	0	0	1
1	0	1	1	1	0	1	0
1	1	0	0	1	0	1	1
1	1	0	1	1	1	0	0
1	1	1	0	1	1	0	1
1	1	1	1	1	1	1	0

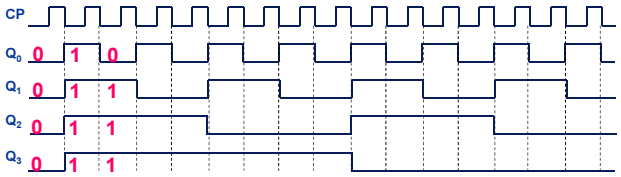
异步二进制减法计数器的状态转换图

（3）状态转换图 $Q_3Q_2Q_1Q_0$



异步二进制减法计数器时序图和电路特点

（4）时序图



- 第1个CP下降沿到来时， Q_0由0变为1，Q_0的上升沿又使Q_1翻转，Q_1的上升沿又使Q_2翻转， Q_2的上升沿又使Q_3翻转，则Q_3 Q_2 Q_1 Q_0从0000变为1111；
- 第2个CP下降沿到来时， Q_0由1变为0， Q_0没有上升沿，则Q_1保持不变，同理， Q_2、Q_3保持不变，故$Q_3Q_2Q_1Q_0$从1111变为1110——减计数。

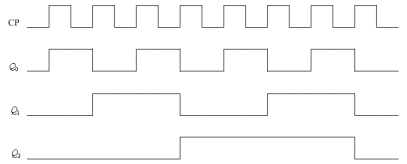
（5）电路特点

异步二进制（M=16）减法计数器

其他类型异步计数器

【例】 D触发器（上跳沿触发）构成的异步二进制（模8）加法计数器

（1）根据题意画出加法计数器时序图



（2）根据时序图写出状态方程

$$Q_0^{n+1} = \overline{Q_0}^n \cdot CP\uparrow$$
$$Q_1^{n+1} = \overline{Q_1}^n \cdot Q_0\downarrow$$
$$Q_2^{n+1} = \overline{Q_2}^n \cdot Q_1\downarrow$$

（3）根据特性方程到驱动方程

$$D_0 = \overline{Q_0}^n \cdot CP\uparrow$$
$$D_1 = \overline{Q_1}^n \cdot Q_0\downarrow$$
$$D_2 = \overline{Q_2}^n \cdot Q_1\downarrow$$

其他类型异步计数器

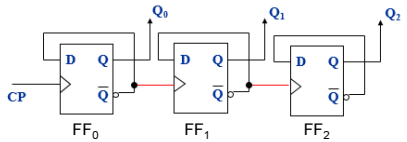
D触发器（上跳沿触发）构成异步二进制（模8）加法计数器（续）

（4）根据驱动方程画出逻辑电路图

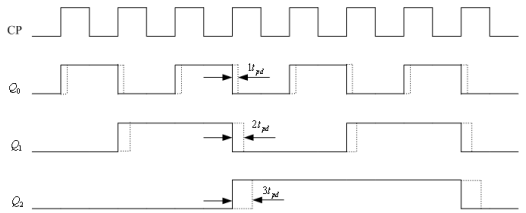
由3个D-FF（D触发器）构成，异步计数器FF_0用CP时钟触发，FF_1用FF_0输出触发，FF_2用FF_1输出触发。

假定D触发器CP上升沿触发，而在驱动方程中FF_1、FF_2分别是在Q_0的下降沿和Q_1的下降沿翻转，也即在/Q_0的上升沿和/Q_1的上升沿翻转。故分别将/Q_0和/Q_1作为FF_1、FF_2的时钟信号，由此画出电路图。

$$D_0 = \overline{Q_0}^n \cdot CP\uparrow$$
$$D_1 = \overline{Q_1}^n \cdot Q_0\downarrow$$
$$D_2 = \overline{Q_2}^n \cdot Q_1\downarrow$$



异步二进制加法计数器的实际时序图



- 考虑各触发器的传输延迟时间t_{pd}时，一个n 位二进制异步计数器，从一个计数脉冲（设上升沿触发）到来，到n 个触发器都翻转稳定，需要经历的最长时间是nt_{pd}。
- 为保证计数器的状态能正确反映计数脉冲的个数，下一个计数脉冲必须在nt_{pd} 后到来，因此计数脉冲的最小周期$T_{min}=nt_{pd}$。

n位二进制异步计数器总结

- n位二进制异步计数器由n个处于计数工作状态（对于D 触发器，使$D_i=\overline{Q_i}$；对于JK 触发器，使$J_i=K_i=1$）的触发器组成。各触发器之间的连接方式由加、减计数方式及触发器的触发方式决定。
  - 对于加计数器，若用上升沿触发的触发器组成，则应将低位触发器的Q非端与相邻高一位触发器的时钟脉冲输入端相连（即进位信号应从触发器的Q非端引出）；若下降沿触发，则应将低位触发器的Q端与相邻高一位触发器的时钟脉冲输入端连接。
  - 对于减计数器，各触发器的连接方式则相反。若触发器上升沿触发，则应将低位触发器的Q端与相邻高一位触发器的时钟脉冲输入端相连；若下降沿触发，则应将低位触发器的Q非端与相邻高一位触发器的时钟脉冲输入端连接。
- 在二进制异步计数器中，高位触发器的状态翻转必须在低一位触发器产生进位信号（加计数）或借位信号（减计数）之后才能实现。故又称这种类型的计数器为串行计数器。也正因为如此，异步计数器的工作速度较低。

2、用反馈复位法实现异步M制计数器

- 异步二进制的计数器电路简单，它的模值是2^n，如果不能改变这个模值，则使用范围就要受到限制。
- 反馈复位法可以改变计数器的模值，得到任意模值的计数器
- 反馈复位法原理：当计数器计到规定的模值时，将计数器为“1”的输出送至反馈电路，产生置0信号/R_D使计数器复位，完成一次计数循环。
- 反馈复位法实现步骤：

  （1）根据计数器模值，求反馈复位代码S_M，即计数器模值的二进制代码；

  （2）求反馈复位逻辑：将输出为1的FF的Q端信号进行逻辑乘后取反，作为反馈复位信号；$\overline{R_D}=\Pi Q^1$

  （3）画逻辑图（先画出由N级FF构成的异步二进制计数器，然后加入反馈复位逻辑）。

用反馈复位法设计异步十进制加法计数器

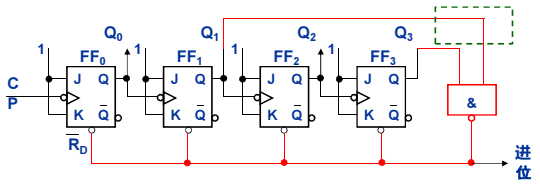
【例】用反馈复位法设计异步十进制加法计数器。

（1）求反馈复位代码 $S_M=(10)_{10}=(1010)_2 (N=4即Q_3Q_2Q_1Q_0)$

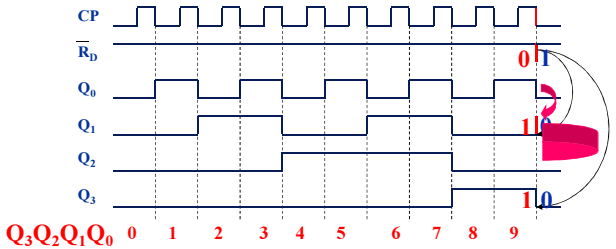
（2）求反馈复位逻辑 $\overline{R_D}=\Pi Q^1=\overline{Q_3Q_1}$

（3）画逻辑图

反馈复位电路



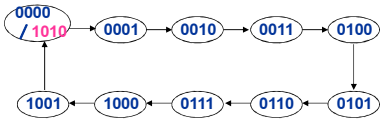
异步十进制加法计数器时序图



- 当第10个CP下降沿到来时，计数器进入1010状态，$Q_3Q_1=11$使/$R_D=0$，计数器复位，$Q_3Q_2Q_1Q_0$变为0000；$Q_3Q_1=00$又使/R_D变为 1——1010状态和/$R_D=0$只出现了瞬间
- 1010状态称为过渡状态，它与0000状态占用一个时钟周期，所以把它们合并在一起。

异步十进制激发计数器状态转换图

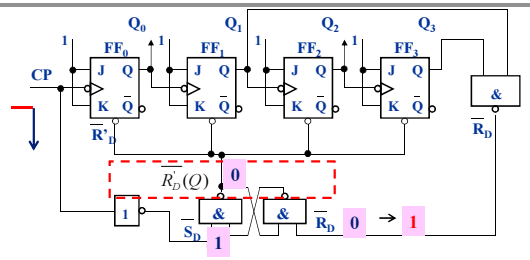
$Q_3Q_2Q_1Q_0$
1010为过渡状态



状态转换图

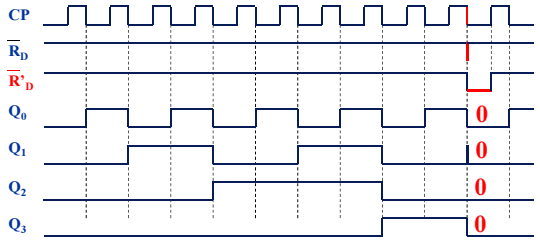
- 上述反馈复位电路存在问题：/R_D的作用时间非常短暂，可能不会使全部FF复位，达不到反馈复位的目的
- 改进：增加基本RS触发器，增加/R_D的作用时间

反馈复位改进电路



- 当CP为1时，/S_D变为0，基本RS触发器置1，计数器不复位。
- 当第10个CP下降沿到来时（则/S_D＝1），计数器进入1010状态，Q_3Q_1=11使/R_D=0，故基本RS触发器置0(/R_D'=0)，使得计数器复位。
- 当/R_D信号撤销后(即/R_D＝1），此时CP＝0，/S_D为1，则基本RS触发器保持不变(/R_D'=0)，从而保证计数器能够可靠复位。

反馈复位改进电路的时序图



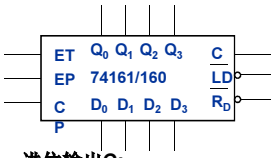
$Q_3Q_2Q_1Q_0$ 0 1 2 3 4 5 6 7 8 9

- 当CP为1时，/S_D变为0，基本RS触发器置1 (/R_D'=1)。
- 当第10个CP下降沿到来时（即/S_D＝1），计数器进入1010状态，Q_3Q_1=11使/R_D=0，则基本RS触发器置0 (/R_D'=0)。当/R_D信号撤销后，此时CP＝0，/S_D为1，基本RS触发器保持0不变(/R_D'=0)，从而保证计数器能够可靠复位。
- 只有当CP上升沿再次到来时，/R_D'才从0变为1。/R_D'的低电平时间等于时钟的低电平时间。

集成计数器

- 集成计数器包括异步和同步计数器。异步计数器有二进制、十进制及可变进制异步计数器。同步计数器包括加法、减法、 加/减（可逆）二进制、十进制同步计数器。
- 4位同步二进制加法计数器74161（异步清除）、74163（同步清除），同步十进制加法计数器74160 （异步清除）、74162（同步清除）

逻辑符号（74161与74160相同）



进位输出C:

74161：$C = ET \cdot Q_3^n \cdot Q_2^n \cdot Q_1^n \cdot Q_0^n$

74160：$C = ET \cdot Q_3^n \cdot Q_0^n$

74161与74160的功能表

$\overline{R_D}$	LD	EP	ET	CP	功能
0	×	×	×	×	异步复位
1	0	×	×	↑	同步预置
1	1	0	0	↑	保持
1	1	0	1	↑	保持
1	1	1	0	↑	保持
1	1	1	1	↑	计数

- EP、ET：功能控制端，为11时计数器计数，其他情况下计数器保持。用于多个计数器的级联

集成计数器实现M进制计数

❖假如一个计数器模值为M，则两片级联后，模值变为M^2。但有时需要的计数器模值不是M^2，例如60进制、24进制

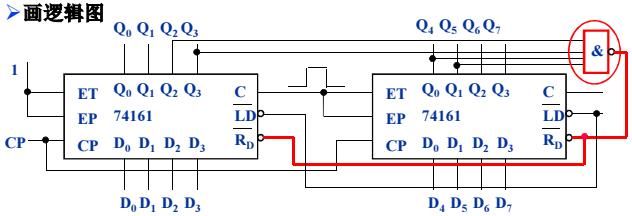
❖反馈复位法或预置法，可以得到任意进制计数器

【例】用74161（二进制）实现M=60的加法计数器

➢反馈复位代码 $S_M=(60)_{10}=(111100)_2(N=6即Q_5Q_4Q_3Q_2Q_1Q_0)$

➢反馈复位逻辑 $\overline{R_D}=\overline{\Pi Q^1}=\overline{Q_5Q_4Q_3Q_2}$

➢画逻辑图



（1）输出C预置法

❖预置法——利用计数器的预置功能，改变计数器的模值，得到任意进制计数器。

❖包括输出C预置法和输出Q预置法。

（1）输出C预置法

将输出C经反相后送预置端/LD；将计数器的预置数据输入端D3~D0接好计算得到的预置数据

➢$\overline{LD}=\overline{C}$即将进位输出C经反相后送$\overline{LD}$端

➢$(预置数据)_2$=(计数器的模值)-(改变后的模值M)

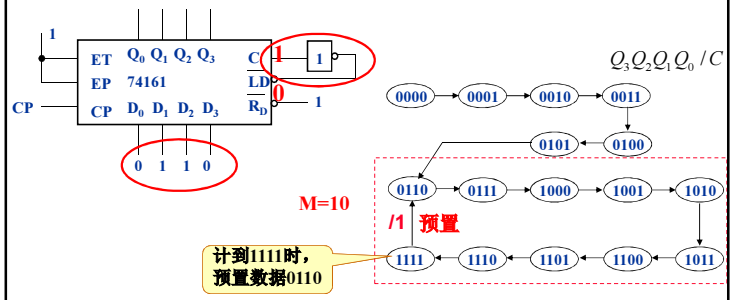
❖ 当计数器未计到最大值时，C=0，则/LD=1，计数器为计数方式；

❖ 当计数器计到最大值时，C=1，则/LD=0，计数器为预置方式，CP到来时，则计数器结束本次计数循环，打入预置数据，并开始下一轮循环。

输出C预置法举例

【例】采用输出C预置法用74161实现M=10加法计数器

$(预置数据)_2=(16-10)_{10}=(6)_{10}=(0110)_2=D_3D_2D_1D_0$



（2）输出Q预置法

（2）输出Q预置法

利用计数器的Q输出接至预置端也可以改变计数器的模值

❖步骤：

① 根据计数器的新模值，求预置代码S_{M-1}，即（计数器新模值-1）的二进制代码；

② 求预置逻辑：当计数器计到（新模值-1）时，将输出为1的FF的Q端信号进行逻辑乘后取反，作为预置信号；$\overline{LD}=\Pi(M-1)^1$

③ 画逻辑图（使并行数据输入即预置数据输入信号恒等于0。）

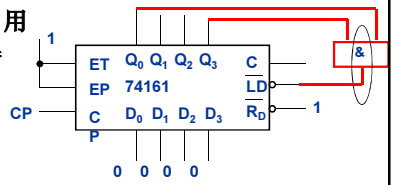
$(预置数据)_2$=0000

输出Q预置法举例

【例】采用输出Q预置法用74161实现M=10加法计数器

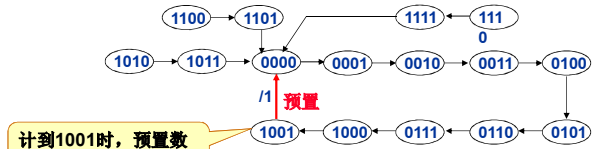
① 求预置代码$S_{M-1} = S_{10-1}=(1001)_2$

② 求预置逻辑　　③ 画逻辑图



$$\overline{LD} = \overline{\prod Q^1} = \overline{Q_3 Q_0}$$

$(预置数据)_2$=0000 = $D_3D_2D_1D_0$　　$Q_3Q_2Q_1Q_0 / C$



第三部分：时序逻辑电路设计

一. 锁存器和触发器
1. SR/D锁存器
2. D触发器
3. 寄存器

二. 有限状态机
1. Moore型有限状态机
2. Mealy型有限状态机

三. 时序逻辑电路设计分析
1. 数据寄存器/锁存器
2. 移位寄存器
3. 计数器
4. Verilog HDL 设计
5. 时序电路的时序

数据寄存器的设计

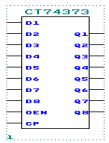
CT74373的设计

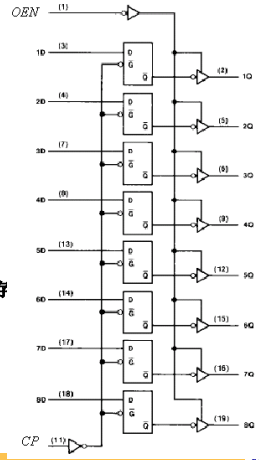
CT74373是带三态输出的锁存器

【例】用always块语句设计8D锁存器

CT74373

- 并行数据输入端：D8-D1，8位；
- 时钟输入端：CP，上升沿有效；
- 并行数据输出端： Q8-Q1， 8位
- 三态控制输入端：OEN，低电平有效，当OEN=0时，锁存器工作；当OEN=1时，锁存器被禁止，输出为高阻态。



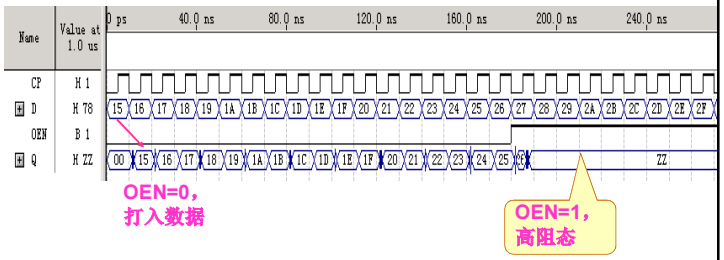


CT74373的HDL设计

```
module CT74373(D1,D2,D3,D4,D5,D6,D7,D8,OEN,CP,
                Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8);
  input   D1,D2,D3,D4,D5,D6,D7,D8,OEN,CP;
  output  Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8;
  reg     Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8;
  reg[1:8] Q_TEMP;
  always @(posedge CP)   //数据暂存到Q_TEMP中
    Q_TEMP = {D1,D2,D3,D4,D5,D6,D7,D8};
  always @(OEN)
    begin
      if (!OEN) {Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8} = Q_TEMP;//锁存器工作
      else {Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8} = 8'bzzzzzzzz; ;//锁存器为高阻态
    end
endmodule
```

❓ ❖ 思考：能否用一个always块表示?

CT74373的仿真波形图



移位寄存器的设计

【例】用always块语句设计8位双向移位寄存器

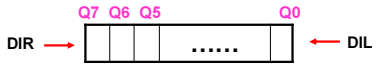
功能：同步复位；同步预置；串入左移；串入右移

信号：d[7..0]是预置输入数据信号；clk是时钟信号，上升沿触发；

clr是复位控制输入端，低有效；lod是预置控制输入端，高有效；

s是移位方向控制输入端，s=1时右移，s=0时左移；

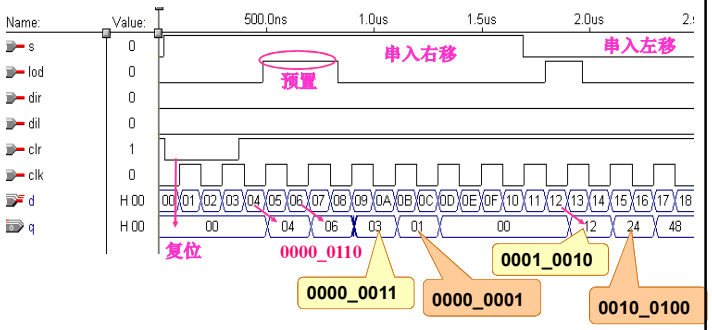
dir是右移串入输入信号；dil是左移串入输入信号。



移位寄存器的HDL设计

```
module rlshift(q,d, clk,clr,lod, s,dir,dil);
    input [7:0]   d;
    input          clk, clr, lod, s, dir, dil;
    output [7:0] q;
    reg  [7:0]     q;
    always @(posedge clk) // 沿触发
      begin
        if (!clr)      q = 8'b00000000;//同步复位
        else if (lod)              q = d;      //同步预置
        else if (s) begin
          q = q >> 1;                    //实现右移操作
          q[7] = dir; end            //寄存器的最高位接收串行右移输入信号
        else  begin
          q = q << 1;                    //实现左移操作
          q[0] = dil; end            //寄存器的最低位接收串行左移输入信号
      end
endmodule
```


8位双向移位寄存器的仿真波形

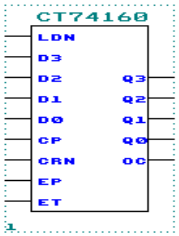


计数器的设计

1、十进制同步计数器（异步清除）CT74160

【例】用always块语句设计十进制同步计数器CT74160

- 并行数据输入端：D3-D0
- 时钟输入端：CP，上升沿有效
- 状态输出端：Q3-Q0，Q3、Q2、Q1和Q0的权值依次为2^3、2^2、2^1和2^0
- 异步复位输入端：CRN，低电平有效
- 同步预置控制输入端：LDN，低电平有效
- 使能控制输入端：EP和ET，高电平有效，均为1时计数器工作，只要有1个信号为0，则计数器保持
- 进位输出端: OC, Q3Q2Q1Q0=1001且ET=1时，OC=1



功能：异步复位；同步预置；计数；保持

CT74160的HDL设计

```
module  CT74160(LDN,D3,D2,D1,D0,CP,CRN,EP,ET,Q3,Q2,Q1,Q0,OC);
  input  LDN,D3,D2,D1,D0,CP,CRN,EP,ET;
  output  Q3,Q2,Q1,Q0,OC;
  reg  Q3,Q2,Q1,Q0,OC;
  reg[3:0]  Q_TEMP;
  always@(posedge CP or negedge CRN )
    begin
      if (!CRN) Q_TEMP = 4'b0000;                    //异步复位
      else
        begin
          if (!LDN) Q_TEMP = {D3,D2,D1,D0};          //同步预置
          else if (EP && ET)                         //计数
            if (Q_TEMP < 4'b1001) Q_TEMP = Q_TEMP + 1;
            else  Q_TEMP = 4'b0000;
          else Q_TEMP = Q_TEMP; //保持
        end
    end
```

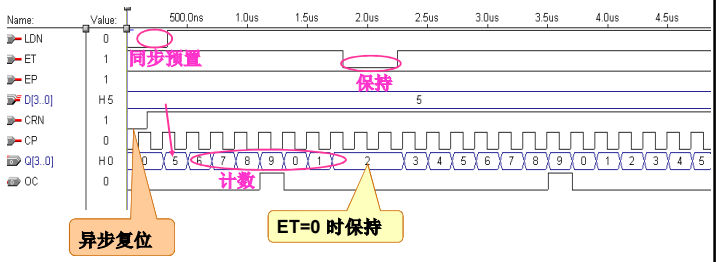
异步复位

最大只计到9

CT74160的HDL设计（续）

```
always                                  //产生进位输出和对最终输出赋值
   begin
      if (Q_TEMP == 4'b1001 && ET == 1'b1) OC = 1'b1;
      else        OC = 1'b0;
      {Q3,Q2,Q1,Q0} = Q_TEMP;
   end
endmodule
```


CT74160的仿真波形图

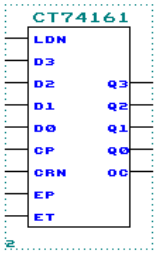


二进制同步计数器CT74161(异步清除)

4位二进制同步计数器（异步清除）CT74161

【例】用always块语句设计二进制同步计数器CT74161

- 并行数据输入端：D3-D0
- 时钟输入端：CP，上升沿有效
- 状态输出端：Q3-Q0， Q3、Q2、Q1和Q0的权值依次为2^3、2^2、2^1和2^0
- 异步复位输入端：CRN，低电平有效
- 同步预置控制输入端：LDN，低电平有效
- 使能控制输入端：EP和ET，高电平有效，均为1时计数器工作，只要有1个信号为0，则计数器保持
- 进位输出端：OC, Q3Q2Q1Q0=1111且ET=1时，OC=1



CT74161的HDL设计

```
module  CT74161(LDN,D3,D2,D1,D0,CP,CRN,EP,ET,Q3,Q2,Q1,Q0,OC);
    input       LDN,D3,D2,D1,D0,CP,CRN,EP,ET;
    output      Q3,Q2,Q1,Q0,OC;
    reg          Q3,Q2,Q1,Q0,OC;
    reg[3:0]    Q_TEMP;
    always@(posedge CP or negedge CRN )
      begin
        if (!CRN) Q_TEMP = 4'b0000;                 //异步复位
        else
          begin
            if (!LDN) Q_TEMP= {D3,D2,D1,D0}; //同步预置
            else if (EP && ET) Q_TEMP = Q_TEMP + 1; //计数
            else Q_TEMP= Q_TEMP; //保持
          end
      end
```

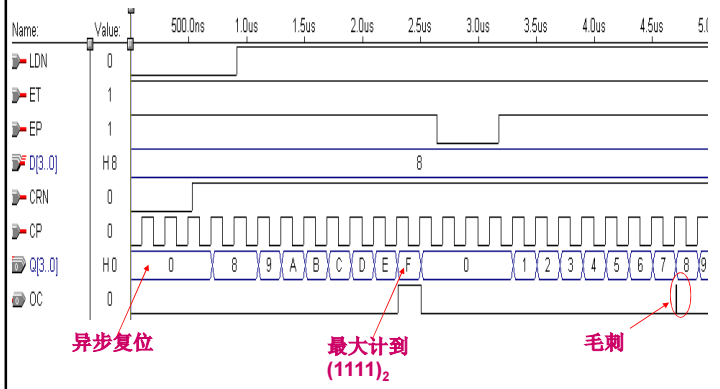
与CT74160的区别:不必判断计数最大值为多少，每来一个时钟脉冲，都加1计数

CT74161的HDL设计（续）

```
always//产生进位输出和对最终输出赋值
        begin
            if (Q_TEMP == 4'b1111 && ET == 1'b1) OC = 1'b1;
            else      OC = 1'b0;
           {Q3,Q2,Q1,Q0} = Q_TEMP;
        end
endmodule
```

与CT74160的区别

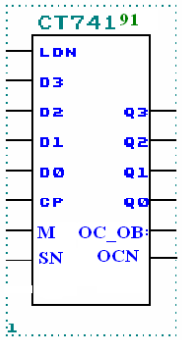
CT74161的仿真波形图



同步加/减计数器CT74191

【例】4位二进制同步加/减计数器CT74191

- 并行数据输入端：D3-D0
- 时钟输入端：CP，上升沿有效
- 状态输出端： Q3-Q0， Q3、Q2、 Q1和Q0的权值依次为2^3、2^2、2^1和2^0
- 加/减控制输入端：M，M=0时，计数器加计数；M=1时，计数器减计数
- 使能控制输入端：SN，低电平有效
- 同步预置控制输入端：LDN，低电平有效
- 进位/借位输出端：OC_OB，加法计数时，当Q3Q2Q1Q0=1111时，OC_OB=1；减法计数时，当Q3Q2Q1Q0=0000时，OC_OB=1；
- OC_OB的反相输出端: OCN，输出脉冲宽度为半个时钟周期（低电平段）



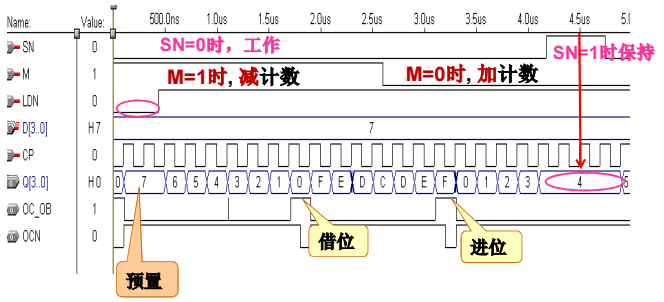
CT74191的HDL设计

```
module CT74191(LDN,D3,D2,D1,D0,CP,M,SN,
                    Q3,Q2,Q1,Q0,OC_OB,OCN);
   input     LDN,D3,D2,D1,D0,CP,M,SN;
   output    Q3,Q2,Q1,Q0,OC_OB,OCN;
   reg       Q3,Q2,Q1,Q0,OC_OB,OCN;
   reg[3:0]  Q_TEMP;
   always @(posedge CP ) //（1）处理加/减操作
     begin
        if (!LDN) Q_TEMP = {D3,D2,D1,D0};    //同步预置
        else if (!SN)                        //计数器工作
          if (!M)  Q_TEMP = Q_TEMP + 1;  //若M为0，做加法
          else Q_TEMP = Q_TEMP - 1;      //若M为1，做减法
        else Q_TEMP = Q_TEMP;                //计数器保持
     end
```


CT74191的HDL设计（续）

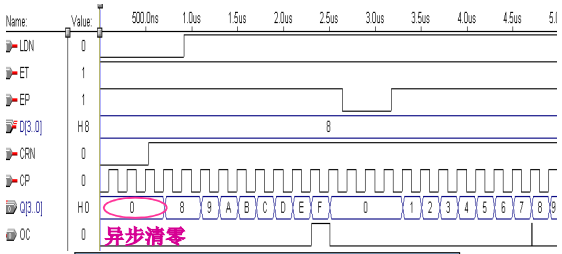
```
always  //（2）处理进位/借位操作
    begin
       if (Q_TEMP == 4'b1111 && M == 1'b0)
         begin
            OC_OB = 1'b1;                    //产生进位
            OCN = (OC_OB && CP);
         end
       else if (Q_TEMP == 4'b0000 && M == 1'b1)
         begin
           OC_OB = 1'b1;                     //产生借位
            OCN = (OC_OB && CP);
         end
       else  OC_OB = 1'b0;
       {Q3,Q2,Q1,Q0} = Q_TEMP;
    end
endmodule
```


CT74191的仿真波形图



计数器同步清零与异步清零

❖ 通常采用异步清零——只要清零信号有效，则无论有无时钟脉冲到来，计数器输出立刻被清零。




```
always @ (posedge CP or negedge CRN)
  begin
    if (!CRN)              //异步清零
      Q<=4'b0000;
    ……
  end
```

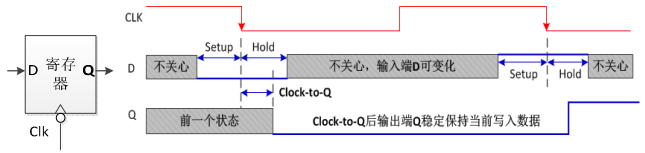

第三部分：时序逻辑电路设计

时序电路的时序

照相机拍照的孔径时间（aperture time）内，被照对象必须保持稳定，否则不能拍到清晰的照片。

时序电路的时序

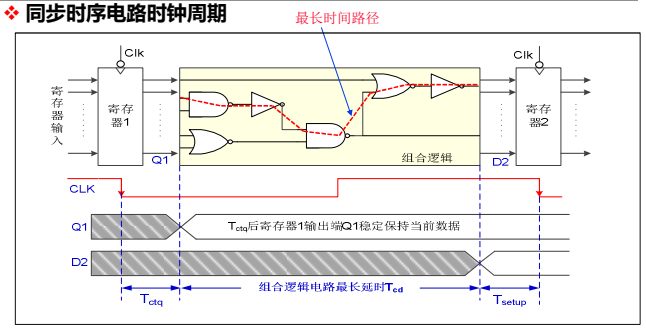
❖寄存器的时序



- 建立时间T_{setup}（Setup Time）：触发时钟边沿之前输入必须稳定的时间；
- 保持时间T_{hold}（Hold Time）：触发时钟边沿之后输入仍需稳定的时间；
- Clock-to-Q时间 T_{ctq}：从触发时钟边沿到输出稳定的时间。
- 孔径时间= T_{setup}+ T_{hold}，输入信号在孔径时间内必须稳定不变。

时序电路的时序

❖ **同步时序电路时钟周期**

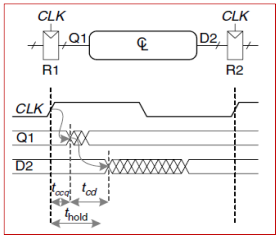


- 时序电路由“ 寄存器元件+ 操作元件(组合电路) + 寄存器元件” 组成
- 时钟周期$T_C >= T_{ctq} + T_{cd} + T_{setup}$ + 时钟偏移
- 时钟偏移：由于时钟信号源到各个寄存器部件的连线长度不同等原因所引起的各个寄存器时钟信号达到的细微时间差异， 一般忽略。

时序电路的时序

❖ **保持时间约束**

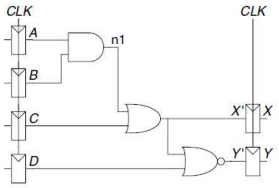
- 右边电路图中寄存器R2有保持时间约束，在时钟上升沿之后的t_{hold}之内，D2必须保持稳定不变。
- 时钟上升沿之后t_{ccq}时，R1的输出Q1将发生变化（但不一定稳定），这种变化经过组合逻辑电路延迟t_{cd}后传播到D2。
- 保持时间约束为$t_{ccq} + t_{cd} >= t_{hold}$，否则R2不能正确对上一个时钟周期内所形成D2的输入，因为在上一个D2的保持时间内，D2就发生了变化。



时序电路的时序

【例】假定下面的电路触发器时钟到Q的最小延迟和稳定时间分别为30ps和80ps，建立时间和保持时间分别为50ps和60ps，每个门电路的最小延迟和最大延迟分别是25ps和40ps。该时序电路的最小时钟周期是多少？并对时序电路进行时序分析。

解：



（1）T_{ctq}=80, T_{setup}=50

组合逻辑最大延迟T_{cd}=3*40=120。

所以最小时钟周期为:

$TC=T_{ctq}+T_{cd}+T_{setup}$

= 80 + 120 + 50 = 250ps

（2）T_{ccq}=30, T_{hold}=60

组合逻辑最小延迟为25（D的改变最快经过25ps可能引起Y'改变）。

T_{ccq}+25=55<T_{hold} 违背了保持约束，Y'值（X'值也同样有可能）不能保持足够长的稳定时间，所以Y值实际上不可预测。因此，该电路在任何时钟周期下其功能都可能不正确。